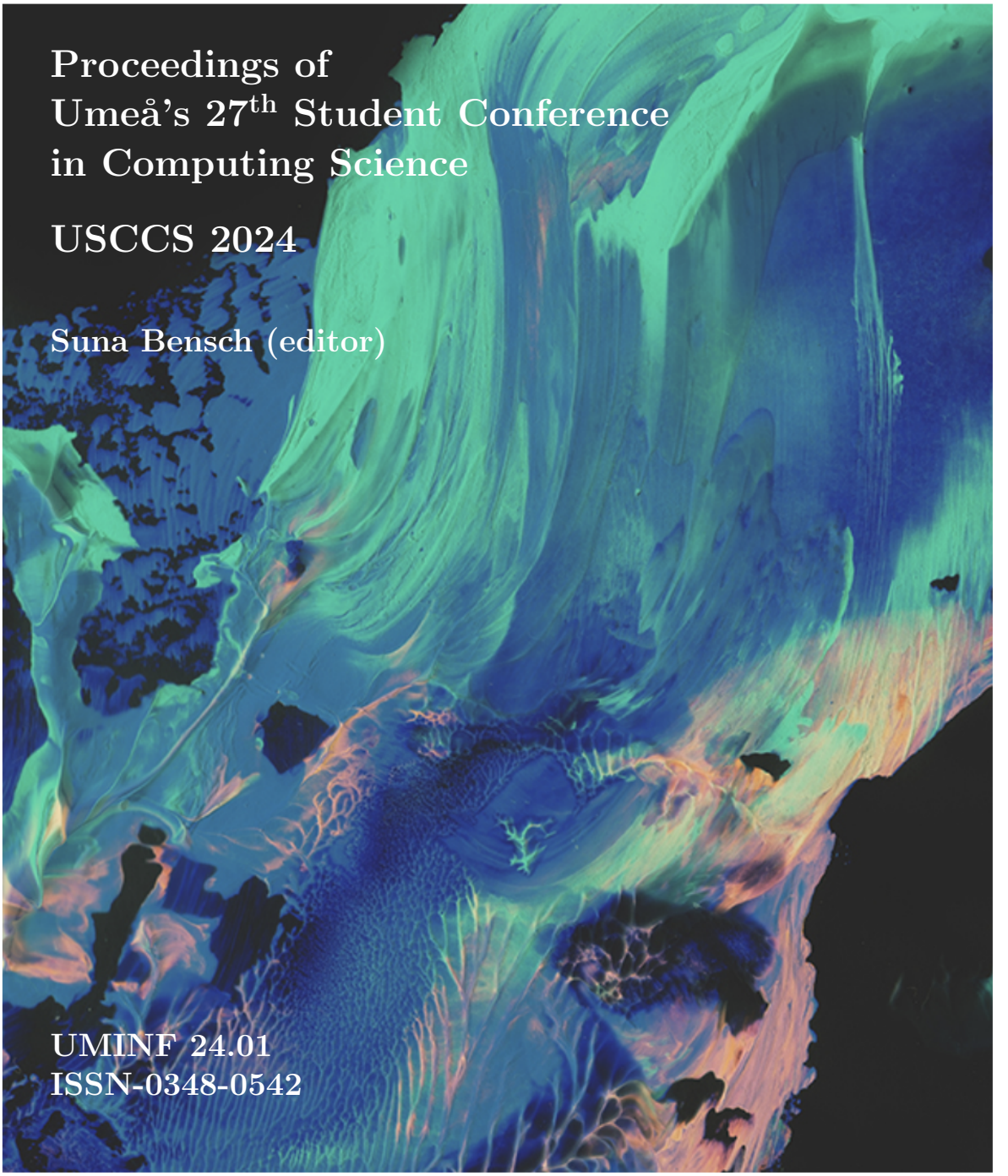
UMEÅ UNIVERSITY

Proceedings of Umeå's 27th Student Conference in Computing Science

USCCS 2024

Suna Bensch (editor)



UMINF 24.01
ISSN-0348-0542

Department of Computing Science
Umeå University

Preface

The Umeå Student Conference in Computing Science (USCCS) is an annual event organized as part of a course offered by the Department of Computing Science at Umeå University. The primary aim of the course is to provide students with a hands-on introduction to independent research, scientific writing, and oral presentation.

A student who participates in the course selects a topic in computing science and related areas and formulates a research question. The course revolves around three significant milestones. The first milestone requires students to write a literature overview with an annotated bibliography, demonstrating not only their academic proficiency but also grounding their research into existing literature - standing on the shoulder of giants. The second milestone involves the actual research and its description in a scholarly manner, demonstrating a commitment to academic excellence, rigour and adherence to high standards. The third milestone encompasses the analysis and discussion of the obtained results, ensuring a thorough and objective examination.

These three milestones are supported by three peer-review group meetings, consisting of 4-5 students each. During these sessions, each ongoing draft or milestone is efficiently and critically discussed aiming at guidance and improvement of the draft. This process provides valuable training in both giving and receiving constructive criticism.

In addition, four lectures support the students' learning and progress in the incremental development and refinement of a scientific paper, and timely discusssions on research ethics and quality.

Each scientific paper is submitted to USCCS through EasyChair, an online submission system, and receives anonymous reviews from experts in the field. Based on the reviews and the editor's assessment a decision of acceptance is made. Reviewers' comments are incorporated, and the revised manuscripts undergo a final review before being included in these proceedings. The review process and conference format aim to simulate realistic settings for publishing processes and participation in scientific conferences.

The conference is the highlight of the course, and this year, we received 14 submissions out of a possible 16, each thoroughly reviewed by experts listed on the following page. As a result, 8 submissions have been accepted for presentation at the conference. We extend our gratitude to the reviewers for their efforts within a tight timeframe and busy schedules.

We also thank all authors for their dedication and outstanding final results, which will be presented during the conference. We wish all participants interesting exchange of ideas and stimulating discussions throughout USCCS.

Umeå, January 2024 Suna Bensch

Organizing Committee

Suna Bensch

With special thanks to the reviewers

Atakan Aral
Jerry Eriksson
Thomas Hellström
Stefan Johansson
Anna Jonsson
Timotheus Kampik
Nina Khairova
Polina Kurtser

Table of Contents

Event-based backpropagation in a discretised spiking neural network framework

Ayush Bahuguna

Umeå University, Department of Computing Science
SE-901 87, Umeå, Sweden
mai22aba@cs.umu.se

Abstract. Spiking neural networks (SNNs) are time-domain models of biophysical dynamics in neurons, that have been shown to be capable of intelligence and the potential to utilise low-power neuromorphic hardware. However, current implementations for optimising spiking networks rely heavily on hardware acceleration which both makes them prohibitive to experimentation and compatitibility with the neuromorphic paradigm. This paper documents an attempt to extend prior methods for parametric optimisation of SNNs by developing a discretised spike response model with the aim of deriving computationally efficient frameworks for learning in neuron models.

1 Introduction

Spiking neural networks (SNNs) are a biologically inspired model of neuronal dynamics [30], used to simulate logical computation in the brain and evaluate biologically plausible theories of intelligence. But SNN models can do more than simulate brains - in recent decades, there has been an effort to combine this blueprint with methods derived from machine learning, where algorithms acquire intelligent decision-making capabilities by being evaluated and optimised on curated data. SNNs have been shown capable of supervised learning for classification [34], [27] and regression [11], [1], as well as unsupervised learning for image reconstruction and classification [4], [32]. This effort has been motivated by the successful application of artificial neural networks (ANNs) in several computational tasks with an adaptibility and generalisability that makes them a potent topic for research in different fields. While an artificial neuron in a feed-forward ANN processes weighted real values as inputs to generate a real-valued output, a spiking neuron processes discrete events over weighted synapses to generate events as output known as *spikes*. ANNs can be considered as quantitative time-averaged SNNs, with the latter being more suitable for probing micro-scale dynamics in neural models. For this reason, SNNs have been considered as platforms for augmenting brain-computer interfaces [26], [14] which are devices designed to facilitate processing and interpretation of electrical activity in biological neural systems.

Learning in spiking neural networks is a complex problem to solve. There is potential insight into *neural plasticity*, how neural networks encode and learn

Suna Bensch (Eds.): Umeå's 27th Student Conference in Computing Science USCCS 2024, pp. 1–14, January 2024.

physically. A breakthrough will have far-reaching consequences for our understanding of brains as dynamic control systems. Advances in neuromorphic computing and memory [24], [31], [17] also motivate study of SNNs which could maximally utilise a paradigm of low-power, low-latency hardware in artificial intelligence applications of different scales; traditionally, computational requirements for deep learning have been prohibitive to widespread utilisation in such paradigms as their design often relies on accelerated computation on contiguous memory representations. A similar avenue of research is the utilisation of memristive technologies [15], which use devices that mimic the behaviour of the theoretical memristor with the capability to modulate circuital resistance based on past electrical activity, effectively providing a physical implementation for memory. It has been suggested that SNN-based intelligent systems can be fashioned that are well suited for exploiting memristor dynamics [6], [2].

Backpropagation [16] is a successful method for optimising learning algorithms with several interdependent parameters by approximating global gradients for estimating effects of collective parametric changes on model behavior by utilising local relationships. Owing to the discrete nature of spike events and how they are represented in SNNs, the use of backpropagation has been avoided in several successful methods for training networks [19], [23] as gradient methods do not apply naturally on systems exhibiting discontinuous states. However, augmented backpropagation techniques have shown a resurgence with demonstrations of competitive performance from gradient-based approaches [13], [33] and the development of surrogate gradient methods [21] that circumvent the discontinuity problem. Recent advances in SNN design have also emphasised the importance of the temporal dynamics that distinguish them from ANNs. Modelling *axonal delays*, the time between generation and incidence of spikes between neurons, has been shown to improve performance [29], and networks which only adopt delay plasticity with highly restricted synaptic weight distributions have been shown to be sufficient for classification tasks [10].

The purpose of this study is to combine the success of efficient backpropagation methods with a computationally efficient framework that allows robust training. The scope is limited here to supervised classification problems, hopefully without affecting the general utility of devised methods for other applications. Although enticing separate field for research, supervised regression and unsupervised learning tasks like image reconstruction have significantly different methods for estimating model performance and hence have different formulations for cost functions, rendering them beyond the scope of this study. Furthermore, the focal point is to utilise discretised implementations of neuron dynamics; differential formulations of spiking neurons have convenient exact solutions however they are computationally expensive. This creates a dependence on hardware acceleration which limits experimentation and reduces applicability of solutions for neuromorphic hardware. Discretising the differential equations provides a straightforward approach to reducing the computational cost of the model with efficient linear expressions but incur instability due to deviations in accuracy of

approximation. Thus, the goal is to probe the applicability of such an approach for deriving efficient alternatives for SNN design.

Section 2 describes related prior work in the field, before Section 3 in which the formulation and proposed adaptation of selected methods is described. Section 4 discusses the results obtained and Section 5 elaborates on conclusions and motivations for future work.

2 Earlier work

A general requirement of time-domain spiking neuron models is to model action potentials which are discharge events in neurons dependent on the *membrane potential* across the cell membrane. This potential is modulated by the simulated currents through ion channels in the cell membrane. These currents originate from the chemical environment of the cell as well as a combination of postsynaptic potentials (PSPs) driven by neurotransmitters from synaptic input sources.

The Hodgkin-Huxley model [12] is among the first recognised biologically accurate formulations for discrete events in neurons. It describes a continuous 4 - dimensional non-linear system modeling the conductance of the cell membrane. The FitzHugh-Nagumo model introduced in [5], [20] is a 2D simplification of the Hodgkin-Huxley model that uses a non-linear membrane potential evolution with a linear potential decay. These continuous dynamical models have been studied intensively for behaviour and stability in phase space. However, due to nonlinearities it is not possible to efficiently compute large populations of neurons in parallel.

In order to study deeply connected neuron populations where linear dynamics can be sufficiently complex, modern approaches adopt *integrate-and-fire* methods where membrane potential is discontinuously modified on crossing a threshold. Leaky integrate-and-fire (LIF) models are a basic formulation which describe membrane potential as linearly integrating input current with exponential decay, while non-linear variants modify the accumulation of current as being dependent on a function of the membrane potential. Gerstner's Spike Response Model (SRM) described in [7] is a seminal work on modeling the behaviour and stability of LIF network models, in which neuron behaviour depends solely on the time since an action potential by the neuron or its input sources.

The complexity of time-domain control systems makes parameter optimisation a challenge. Spike-timing-dependent plasticity (STDP) introduced in [19] is among the first successful formulations of supervised learning for linear LIF models, where weight updates are calculated using a cascading reward dependent on the relative difference between the time of incident PSPs and the time of the postsynaptic neuron's firing. The reward is empirically devised and applied in a supervised manner depending on whether a neuron is expected to fire or not. The Remote Supervised Method (ReSuMe) described in [23] is an algorithm to train neurons to learn target firing patterns driven by deterministic stimuli. As opposed to the empirical, discrete reward approach in STDP, it applies time-

continuous weight modulation dynamics based on a formulation of local error between generated spikes and expected event times.

Both STDP and ReSuMe are designed for optimising single neurons and need to be augmented in order to utilise them for training deep networks. For arbitrarily deep networks, stable optimisation requires estimation of gradients of cost, making an approach based on backpropagation desirable. One algorithm that acheives this successfully is SLAYER, which is described in [25]. The algorithm evaluates the loss at every time step of the simulation for the entire network and is able to derive highly performant models and is also capable of axonal delay optimisation, however this approach is computationally costly. The EventProp adjoint method derived in [33] provides a framework for training a leaky-integrate-and-fire [18] SNN with backpropagation in a manner reminiscent of training in traditional ANNs [16], by assuming a locally linear evolution of membrane potential at the time of spike. Hence it is computationally simpler than SLAYER as it limits the backpropagation process only to the time steps where neurons fire or are expected to fire. This study aims to adapt the EventProp algorithm for updating both synaptic weights and axonal delays for a spiking neural network utilising discretised update operations.

3 Methodology

This section elaborates on the details of the proposed framework. The output of a parametrised model is easily known for any given input state which is often referred to as the *forward problem*, but it is harder to calculate modulations of the parameters in order to achieve a more desireable output, posed as the *inverse problem*. The use of Lagrangian optimisation for deriving model updates to improve the performance of the model is described. Here, an important caveat is that the interdependence of neuronal states in a discretised model is no longer described by exact solutions, hence the gradients of individual states need to be derived such that they correspond to the approximations for neuronal activity introduced by the choice of time discretization. This is further complicated by the fact that spiking networks exhibit discontinuities, which are created at the time of action potentials where neurons sharply reset states. Since the performance of the model is described by the action potentials generated at the output layer as well as the fact that hidden layers are driven by the action potentials generated at preceding sources, optimisation has to be performed by considering the behaviour of the model components near discontinuities independently from other variables.

Subsection 3.1 shows the definition of the architecture for the SNN model and the functionality of its components and Subsection 3.2 describes the mathematical formulation for an optimisation scheme for the models parameters. Subsection 3.3 further expands on how this optimisation scheme was adapted for our discretised framework, followed by Subsection 3.4 which details how the methods are adapted to be robust around discontinuities in system representations

3.1 Spike Response Model

The proposed methodology models a neuron with n input sources as having an exponentially decaying membrane potential E defined as the sum $\epsilon + \eta$ where ϵ represents the integrated input PSPs and η represents an additive refractory kernel which is activated when the neuron generates action potentials. The behaviour of ϵ is described by the differential equation

$$\frac{d\epsilon}{dt} = \frac{-\epsilon}{\tau_s} + \kappa_\epsilon \alpha + I_0 \tag{1}$$

where τ_s is called the *synaptic time constant*, κ_ϵ is the *responsiveness* of the neuron and I_0 is a base temporal activation representing an external driving current referred to as *bias current*. α represents the combined input current from all input sources, which itself is described by an exponential decay process

$$\frac{d\alpha}{dt} = \frac{-\alpha}{\tau_m}, \qquad t = t_i + \delta_i \Rightarrow \alpha := \alpha + w_i \tag{2}$$

where τ_m is called the membrane time constant. The input current is increased by some value when system time exceeds the time t_i when the ith source is activated in addition to the synaptic delay δ_i between the source and the neuron. This value w_i represents the weight of the ith synapse in the activation of the modeled neuron and is called synaptic efficacy.

We similarly model the behaviour of the refractory kernel η using a decay model

$$\frac{d\eta}{dt} = \frac{-\eta}{\tau_s} + \kappa_\eta \beta, \ \frac{d\beta}{dt} = \frac{-\beta}{\tau_m}, \quad t = t_s \Rightarrow \beta = \beta_0 \tag{3}$$

where t_s is the time when $\epsilon + \eta$ exceeds a threshold Θ. β represents a depolarisation current that is negative in magnitude, where the parameter β_0 is the initial depolarisation that is used to control the minimum of η. This minimum is achieved at $t = t_s + \frac{\tau_m \tau_s}{\tau_m - \tau_s} ln(\frac{\tau_m}{\tau_s})$ and has the value $\frac{\nu^{\frac{\nu}{\nu-1}}}{\kappa_\eta(\nu-1)}\beta_0$ where $\nu = \frac{\tau_m}{\tau_s}$. The algorithm updates the state of the neuron on firing with

$$E > \Theta \Rightarrow \epsilon := 0, \alpha := 0, \eta := 0, \beta := \beta_0, t_s := t. \tag{4}$$

Fig. 1 shows the behaviour of the various potential terms for a single neuron with multiple incident action potentials. Non-linear variants of this basic SRM utilise dynamic time constants or thresholds, but since we keep them constant and uniform for this study the only trainable parameters are the synaptic weights and axonal delays between each neuron and its input sources and the bias current of each neuron. Gerstner's LIF SRM uses exact solutions for these decay equations but we define our model using discretised solutions. Primarily this is because the decay of membrane potential for all neurons in a network can be represented as a single linear operation. Our choice of additive refractory kernel being similar to the behaviour of ϵ also means that the contribution to a neuron's PSP from the most recent spike from a preceding neuron can be approximately estimated by the value of η for the preceding neuron. The importance of this for optimisation is discussed in subsection 3.4.

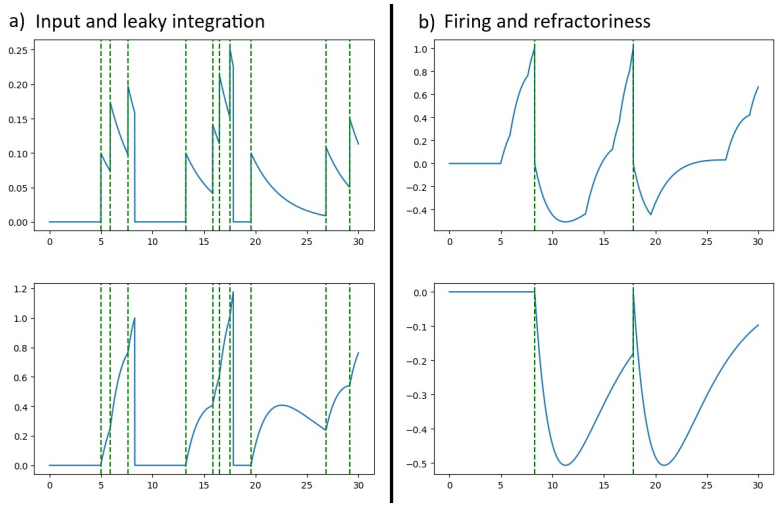


Fig. 1. Leaky integrate-and-fire dynamics. a) α in nA and ϵ in 100mV of a LIF neuron for a spike train (vertical lines denote incident action potentials). b) Corresponding membrane potential E and its refractoriness component η both in 100mV (vertical lines denote time of spike). Notice how ϵ has to be higher to generate the second spike to compensate for transient η.

3.2 Adjoint method

For traditional deep ANNs, backpropagation is clearly the most popular method for estimating cost gradients due to its success, simplicity and adaptability. The concept can be understood effectively by applying the method of Lagrange multipliers considering a Lagrangian function $\mathcal{L}$ incorporating a cost function $\mathcal{C}$ and m constraints $g_1, g_2 \dots g_m$ on the model parameters. Thus,

$$\mathcal{L} = \mathcal{C} + \sum_{r=1}^{m} \lambda_r g_r \tag{5}$$

where the model is constrained such that $g_r = 0 \ \forall \ r \ \in [1, 2 \dots m]$. In the case of ANNs, these g_rs correspond to numerical constraints between layers where the input of one logical unit is dependent on the activations of preceding units. However in the spiking case there are multiple constraints between units as there may be multiple incident action potentials from the same source to a neuron. A solution for a local extremum satisfies the system of equations corresponding to setting all partial derivatives of $\mathcal{L}$ to zero. We set $\frac{\partial \mathcal{L}}{\partial \lambda_{L,i}} = 0$ to enforce the constraint $g_{L,i}$ for the ith action potential incedent layer L and its input sources. Setting $\frac{\partial \mathcal{L}}{\partial t_{r,i}} = 0$ gives relations between the λ_rs where $t_{L,i}$s are action potentials generated at layer L. These relations correspond to the backpropagation equations which allow estimation of gradients for preceding layers. A solution for $\frac{\partial \mathcal{L}}{\partial \theta} = 0$ is often intractable, so we calculate the value of $\frac{\partial \mathcal{L}}{\partial \theta}$ for output layers directly and derive values for hidden layers by backpropagation

using the calculated values of λs. We then minimise these gradients using the gradient descent method by repeatedly modifying every parameter θ using the update equation

$$\theta := \theta - \ell \frac{d\mathcal{C}}{d\theta}, \tag{6}$$

where ℓ is called the *learning rate* or *step size*. This equation can be augmented to make the optimisation process more stable, for example averaging gradients across a batch of datapoints before updating parameters or performing regularisation where the update is limited depending on the magnitude of the parameter to prevent unbounded growth.

Consider a network with layers with N neurons that spike i times. For sake of brevity and readability of the formulation, N and i may be implicitly different for each layer and neuron respectively. If we constrain the layers such that the times a layer receives input are the times $t_{l,r}$, that the preceding neurons fire plus some delay, then potential E_{l-1} at layer $l-1$ must be zero at all times $t_{l,r}\ \forall\ r \in [1 \dots i]$. Hence we get i constraints for each neuron

$$\mathcal{L} = \mathcal{C}(t_{L,0}, \dots\ t_{L,i}) + \sum_{l=1}^{L} \sum_{n=1}^{N} \sum_{r=1}^{i} \lambda_{n,i}^{l} (E_n^l(t_{n,i}^l) - \Theta_i^l) \tag{7}$$

For the output layer L, the gradients depend on the cost and the dynamics of η between consecutive action potentials of the neuron. Since partial derivative of cost is only dependent on time of action potentials generated at the output layer, the dynamics of a hidden layer are determined by its constraints with the next layer

$$\frac{\partial \mathcal{L}}{\partial t_{n,i}^l} = \lambda_{n,i+1}^l \frac{dE_n^l(t_{n,i+1}^l)}{dt_{n,i}^l} + \lambda_{n,i}^l \frac{dE_n^l(t_{n,i}^l)}{dt_{n,i}^l} + \sum_{n=1}^{N} \lambda_{n,i*}^{l+1} \frac{dE_n^{l+1}(t_{n,i*}^{l+1})}{dt_{n,i}^l} = 0 \tag{8}$$

where $i*$ is the implicit index of the next pertinent spike in each neuron of layer $l+1$, the earliest next spike such that $t_{i*}^{l+1} > t_{i-1}^l + \delta^{l+1}$. This spike resets the neuron's variables and thus eliminates any effect $E^l(t_{i-1}^l)$ could have on later action potentials which will be wholly determined by t_{i*}^{l+1}. The term is absent when no such t_{i*}^{l+1} exists. For this same reason, for all $t_j^l > t_{i+1}^l$, $E^l(t_j^l)$ is also independent of t_i^l.

Thus, the dynamics for neurons in hidden layers and the output layer respectively depend on the recurrences

$$\lambda_{n,r}^l = -\left(\frac{dE_n^l(t_{n,r}^l)}{dt_{n,r}^l}\right)^{-1} \left(\lambda_{n,r+1}^l \frac{dE_n^l(t_{n,r+1}^l)}{dt_{n,r}^l} + \sum_{m=1}^{N} \lambda_{m,r*}^{l+1} \frac{dE_m^{l+1}(t_{m,r*}^{l+1})}{dt_{m,r}^l}\right), \tag{9}$$

$$\lambda_{n,r}^L = -\left(\frac{dE_n^L(t_{n,r}^L)}{dt_{n,r}^L}\right)^{-1} \left(\lambda_{n,r+1}^L \frac{dE_n^L(t_{n,r+1}^L)}{dt_{n,r}^L} + \frac{dC}{dt_{n,r}^L}\right)$$

This allows to calculate the λs which allows deriving parametric upgrades with

$$\frac{d\mathcal{L}}{dw_{n,m}^L} = \frac{d\mathcal{C}}{dw_{n,m}^L} + \lambda_{L,r}\frac{dE^L(t_{n,m}^L)}{dw_{n,m}^L} = 0 \tag{10}$$

These λs, which effectively represent the gradients for the backpropagation process, are the adjoint variables of the system. Similar to the EventProp algorithm, they describe an adjoint spiking network which produces parametric updates near action potentials, where in this approach they are separated from neuronal events by axonal delays.

3.3 Discretisation

The expressions for input current α and depolarisation current β in equations 2 and 3 describe straightforward exponential decay. This allows solving the time-dependent ordinary differential expressions in equations 1 and 3 describing the potential terms ϵ and η, with the exact solution of ϵ for a neuron with N inputs taking the form

$$\epsilon = k_\epsilon I_0 \tau_s + \sum_{n=1}^{N}\sum_{r=1}^{i} w_n k_\epsilon \frac{\tau_m - \tau_s}{\tau_m \tau_s}(e^{-\frac{t-t_{n,r}-\delta_n}{\tau_m}} - e^{-\frac{t-t_{n,r}-\delta_n}{\tau_s}}) \tag{11}$$

where $t_{n,i}$ is implicitly the last action potential from input source n and δ_n is the corresponding axonal delay. η has the similar solution

$$\eta = k_\eta \frac{\tau_m - \tau_s}{\tau_m \tau_s}(e^{-\frac{t-t_s}{\tau_m}} - e^{-\frac{t-t_s}{\tau_s}}) \tag{12}$$

where t_s is the last action potential of the modelled neuron. Often, equation 11 is simplified by ignoring the effects of trailing currents from past spikes from input sources leading to the approximation

$$\epsilon = k_\epsilon I_0 \tau_s + \sum_{n=1}^{N} w_n k_\epsilon \frac{\tau_m - \tau_s}{\tau_m \tau_s}(e^{-\frac{t-t_{n,i}-\delta_n}{\tau_m}} - e^{-\frac{t-t_{n,i}-\delta_n}{\tau_s}}) \tag{13}$$

This approximation is preferred in most spike response models because it has the elegant property that the future state of all neurons depends only on the time of last generation of action potentials of all neurons which is a major optimisation of computational cost. Even so, the calculation of exponentials is significantly more expensive than the discretised form which simply requires modifying the state of the neuron with linear operations. For example, by defining the granularity of the time discretisation with a value for Δt, updating ϵ can be derived from equation 1 with the update

$$\epsilon := \epsilon + (\frac{-\epsilon}{\tau_s} + k_\epsilon \alpha + I_0)\Delta t \tag{14}$$

However formulating the derivative with respect to model parameters is more nuanced. For the optimisation procedure for a parameter θ, we also require estimates for $\frac{dE}{d\theta}$. For the case of axonal delays, we can utilise the fact that

$$\frac{dE^l}{d\delta^l} = -\frac{dE^l}{dt}. \tag{15}$$

The derivative with respect to weights only depends on ϵ as η is independent of any synaptic efficacy. In the exact solution from equation 13 we can clearly differentiate the expression and get the individual gradient with respect to a particular weight from the sum. But since in our discretisation we do not have an explicit sum we need to somehow independently model the integrated currents originating from individual input sources. For this we utilise the similarity between the formulation for ϵ and η; we can clearly see from equations 12 and 13 that the current from an input neuron is functionally identical to the negative of the refractory kernel except for a proportionality depending on the weight and a shift in time caused by the axonal delay. Thus the contribution of a neuron N_l at layer l in the value of ϵ at layer $l+1$ is approximated in negative by η at layer l; an additive refractive kernel is a persistent representation of the neuron's contribution to activity at the next depth of the network. This memory is approximate because ϵ_{l+1} might be affected by multiple action potentials from N_l while η_l depends only on the last time N_l spiked. Hence we can approximate derivatives with respect to weights using

$$\frac{dE^l}{dw^l}\Big|_{t=t_s^l} = \frac{-\eta^{l-1}}{\beta_0}\Big|_{t=t_s^l-\delta^l}. \tag{16}$$

3.4 Critical decisions at discontinuity

An obvious complaint with the formulated cost gradients in equation 9 is that when $\frac{dE_n^l}{dt_{n,r}^l}$ vanishes, the gradient is unbounded. However, they can be bounded by exploiting knowledge of the discontinuous behaviour of the model. Since the potential is constrained at each $t_{n,r}^l$, its cooccurrence with a vanishing potential gradient means that the membrane threshold is being exceeded right before a local maximum. In such a state, a small change to model parameters might limit the potential below the threshold, making the constraint $g_{n,r}^l$ invalid and generating a discontinuity in model dynamics. This means that either one has to optimise their model leaving all critical neurons unconstrained, or otherwise derive a constraint that is well-behaved or approximately continuous across the action potential.

For this reason, critical neurons are assigned a relaxed inequality constraint

$$\epsilon^{l+1}(t^l + \delta^{l+1}) + \frac{w^{l+1}}{\beta_0}\eta^l \geq 0. \tag{17}$$

This constraint also utilises the fact that the contribution of an input neuron to the activation of a destination neuron is approximated by $-\eta$ at the input

neuron. It is also necessary that the cost function be modified to exclude all critical neurons and be augmented with a surrogate differentiable loss function for evaluating the critical neurons at discontinuity. Several surrogate gradient methods are described in [21].

4 Results

The implementation details of all tested networks are described as follows:

1. **Parameter distributions**: Xavier weight initialization [8] for weights and bias currents; delays are all initialised uniformly at beginning of learning to 8 time steps where $dt = 0.1$ in the time discretisation. $\tau_m = 3, \tau_s = 0.7, k_\epsilon = 2, k_\eta = 4$
2. **Inhibition**: The model does not implement inhibition of neurons independently, which is instead simulated using negative weights. Neurons with negative synaptic weights induce negative input current and hence inhibit neuron potential from crossing threshold.
3. **Encoding**: The models were tested with phase coding using average cosine loss, where the first spike of an output neuron is considered as the the start of phase and subsequent spikes are evaluated by taking difference in time from first spike. The spike timing is optimised to correlate with the phase of the cost function, where several neurons with different phases were used for each output class.
4. **Surrogate**: Exponential function of membrane potential, similar to SLAYER

The derived approach was first tested with small networks for learning the task of exclusive or (XOR) with small networks and was able to accurately learn the task with a learning rate of 10^{-5} within 1000 iterations. However at higher learning rates the process gets stuck in oscillating updates around local minima of cost.

Larger fully-connected networks were modelled to be trained on handwritten digit recognition with the MNIST dataset [3], where the structure of the networks was adapted from the experiments on the HM2-BP algorithm in [13]. However, most experiments with networks having two hidden layers failed to produce stable results as the backpropogation of errors generates numerical instability with the emergence of *exploding gradients* [22], where cumulatively derived gradients at lower depths become progressively larger causing unstable updates to parameters which compound as training proceeds. When regularisation was introduced into equation 6 with a regularisation rate higher than 10^{-6}, it restricted the training procedure sufficiently and did not produce any meaningful learning in the trained models; lower values of the parameter did not prevent instability in parameter growth. The best performing single hidden-layer model achieved only 68% accuracy which suggests that the numerical instability is still somewhat present in small networks with large layer widths.

The derived method is hence currently unsuitable for training deep networks. Hence the results suggest that it is difficult to reproduce the success of backpropagation in the simultaneous training of weights and delays, and more sophisticated techniques need to be developed in order to create approaches for training spiking neural networks in a computationally efficient manner. It is also possible that the numerical instability originates from the discretisation of the model and it is worth exploring if the algorithm still performs badly when utilising exact solutions of the equations of model dynamics when initialised identically to the proposed method. However, reducing dt further below 0.1 does not achieve any significant improvement to model performance. This suggests that the instability is possibly caused by approximating gradients with respect to model parameters using η from the previous layers like in equation 16. This could potentially be tackled with distinctly modelling the individual contributions of neuron spikes independently which would be incur significant computational and memory costs but this could be limited to the training stage, thus still producing a performant model for inference. There is also the possibility that magnitude of parameter updates is not controlled adequately with large layer widths, which would suggest that there is scope for improvement in minimising the gradient instability by experimenting with different parameter initialisations or batch normalisation.

5 Discussion

The goal of this study was to probe the feasibility of discretisation as a means for computational efficiency in spiking neural networks with weights and delays. The benefit of such a formulation would essentially only be worthwhile for large and deep networks, hence there is a lot of research required before this approach can be made viable. Since the aim was to be "fast and efficient", the EventProp algorithm was chosen as a platform for development due to its simplicity and minimal computation, however the original algorithm was not designed with axonal delays. A more fruitful approach might have been to compromise on cost during training by adapting more complex algorithms which incorporate delays like SLAYER while aiming to produce equivalently efficient discretised models during inference.

Another interesting approach for tackling exploding gradients is to explore regularisation methods specific to the SNN paradigm. As shown in [9], [28], there is evidence that mimicking biological neural processes of sleep by periodically inducing global low frequency stimuli while minimising output response helps the model to maximise its capacity for learning and prevent catastrophic forgetting in visual tasks. This can be considered a form of regularisation where the distribution of model parameters are well-behaved across training over different problems. Since the degree of activity in response to low-frequency stimuli can distinguish exaggerated synaptic efficacy, it can help to identify and resolve exploding gradients if utilised in tandem with the training procedure. Considering the results of this study, it is thus an important consideration for related future work.

By discretising the differential definitions for neuron potentials, the update process for the entire SNN can be represented with linear vector operations, which has to be followed by a discontinuous modifications for spiking neurons whose potential has crossed their threshold in both the discretised and exact formulations. This is a significant speedup as the exact solution requires calculating exponential terms. Introduction of axonal delays also creates avenues for improvement if the delays are sorted by magnitude; if a spike from a neuron has not yet informed its destination through the synapse with shortest axonal delay, then all other synapses can be skipped for that iteration. Since a neuron can spike multiple times before its first action potential has been completely processed, operating on spikes in order of time can also reduce the number of spikes that need to be processed in an iteration. However, since these optimisations require several comparison operations there is poor potential for parallelisation without the employment of independent workers operating on the routing of spike information through the network. If such a formulation could be made stable, there is potential for an asynchronous spiking network paradigm which could have consequences for emergent collective intelligence in distributed cooperating networks. Hence there is motivation to address the failure of the current configuration and attempt to build an improved framework that can yield competetive results.

References

1. Iman AbouHassan, Nikola K. Kasabov, Vinayak Jagtap, and Parag Kulkarni. Spiking neural networks for predictive and explainable modelling of multimodal streaming data with a case study on financial time series and online news. *Scientific Reports*, 13(1):18367, Oct 2023.
2. Aabid Amin Fida, Farooq A. Khanday, and Sparsh Mittal. An active memristor based rate-coded spiking neural network. *Neurocomputing*, 533:61–71, 2023.
3. Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.
4. Yiting Dong, Dongcheng Zhao, Yang Li, and Yi Zeng. An unsupervised stdp-based spiking neural network inspired by biologically plausible learning rules and connections. *Neural Networks*, 165:799–808, 2023.
5. Richard FitzHugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical journal*, 1 6:445–66, 1961.
6. Mohammed E. Fouda, Fadi Kurdahi, Ahmed Eltawil, and Emre Neftci. Chapter 19 - spiking neural networks for inference and learning: a memristor-based design perspective. In Sabina Spiga, Abu Sebastian, Damien Querlioz, and Bipin Rajendran, editors, *Memristive Devices for Brain-Inspired Computing*, Woodhead Publishing Series in Electronic and Optical Materials, pages 499–530. Woodhead Publishing, 2020.
7. Wulfram Gerstner. Time structure of the activity in neural network models. *Phys. Rev. E*, 51:738–758, Jan 1995.
8. Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2010.

9. Ryan Golden, Jean Erik Delanois, Pavel Sanda, and Maxim Bazhenov. Sleep prevents catastrophic forgetting in spiking neural networks by forming a joint synaptic weight representation. *PLOS Computational Biology*, 18(11):1–31, 11 2022.
10. Edoardo Grappolini and Anand Subramoney. Beyond weights: Deep learning in spiking neural networks with pure synaptic-delay training. In *Proceedings of the 2023 International Conference on Neuromorphic Systems*, ICONS '23, New York, NY, USA, 2023. Association for Computing Machinery.
11. Elisa Guerrero, Fernando M. Quintana, and Maria P. Guerrero-Lebrero. Event-based regression with spiking networks. In Ignacio Rojas, Gonzalo Joya, and Andreu Catala, editors, *Advances in Computational Intelligence*, pages 617–628, Cham, 2023. Springer Nature Switzerland.
12. A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4):500–544, 1952.
13. Yingyezhe Jin, Wenrui Zhang, and Peng Li. Hybrid macro/micro level backpropagation for training deep spiking neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 7005–7015, Red Hook, NY, USA, 2018. Curran Associates Inc.
14. Kaushalya Kumarasinghe, Nikola Kasabov, and Denise Taylor. Brain-inspired spiking neural networks for decoding and understanding muscle activity and kinematics from electroencephalography signals during hand movements. *Scientific Reports*, 11(1):2486, Jan 2021.
15. Mario Lanza, Abu Sebastian, Wei D. Lu, Manuel Le Gallo, Meng-Fan Chang, Deji Akinwande, Francesco M. Puglisi, Husam N. Alshareef, Ming Liu, and Juan B. Roldan. Memristive technologies for data storage, computation, encryption, and radio-frequency communication. *Science*, 376(6597):eabj9979, 2022.
16. Yann Lecun. A theoretical framework for back-propagation. In D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School, CMU, Pittsburg, PA*, pages 21–28. Morgan Kaufmann, 1988.
17. Hoo-Cheol Lee, Jungkil Kim, Ha-Reem Kim, Kyoung-Ho Kim, Kyung-Jun Park, Jae-Pil So, Jung Min Lee, Min-Soo Hwang, and Hong-Gyu Park. Nanograin network memory with reconfigurable percolation paths for synaptic interactions. *Light: Science & Applications*, 12(1):118, May 2023.
18. Hendrik M. Lehmann, Julian Hille, Cyprian Grassmann, and Vadim Issakov. Leaky integrate-and-fire neuron with a refractory period mechanism for invariant spikes. In *2022 17th Conference on Ph.D Research in Microelectronics and Electronics (PRIME)*, pages 365–368. IEEE, 2022.
19. Henry Markram, Wulfram Gerstner, and Per Jesper Sjöström. Spike-timing-dependent plasticity: A comprehensive overview. *Frontiers in Synaptic Neuroscience*, 4, 2012.
20. J. Nagumo, S. Arimoto, and S. Yoshizawa. An active pulse transmission line simulating nerve axon. *Proceedings of the IRE*, 50(10):2061–2070, 1962.
21. Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.
22. Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. Understanding the exploding gradient problem. *ArXiv*, abs/1211.5063, 2012.
23. Filip Ponulak and Andrzej Kasiński. Supervised learning in spiking neural networks with resume: Sequence learning, classification, and spike shifting. *Neural Computation*, 22(2):467–510, 2010.

24. Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, Nov 2019.
25. Sumit Bam Shrestha and Garrick Orchard. Slayer: Spike layer error reassignment in time. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.
26. Sai Kalyan Ranga Singanamalla and Chin-Teng Lin. Spiking neural network for augmenting electroencephalographic data for brain computer interfaces. *Frontiers in Neuroscience*, 15, 2021.
27. Christoph Stöckl and Wolfgang Maass. Optimized spiking neurons can classify images with high accuracy through temporal coding with two spikes. *Nature Machine Intelligence*, 3(3):230–238, Mar 2021.
28. Timothy Tadros, Giri P. Krishnan, Ramyaa Ramyaa, and Maxim Bazhenov. Sleep-like unsupervised replay reduces catastrophic forgetting in artificial neural networks. *Nature Communications*, 13(1):7742, Dec 2022.
29. Aboozar Taherkhani, Ammar Belatreche, Yuhua Li, and Liam P. Maguire. Dl-resume: A delay learning-based remote supervised method for spiking neurons. *IEEE Transactions on Neural Networks and Learning Systems*, 26(12):3137–3149, 2015.
30. Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, and Anthony Maida. Deep learning in spiking neural networks. *Neural Networks*, 111:47–63, 2019.
31. Bo Wang, Jun Zhou, Weng-Fai Wong, and Li-Shiuan Peh. Shenjing: A low power reconfigurable neuromorphic accelerator with partial-sum and spike networks-on-chip. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 240–245. IEEE, 2020.
32. Chong Wang, Hongmei Yan, Wei Huang, Wei Sheng, Yuting Wang, Yun-Shuang Fan, Tao Liu, Ting Zou, Rong Li, and Huafu Chen. Neural encoding with unsupervised spiking convolutional neural network. *Communications Biology*, 6(1):880, Aug 2023.
33. Timo C. Wunderlich and Christian Pehle. Event-based backpropagation can compute exact gradients for spiking neural networks. *Scientific Reports*, 11(1):12829, Jun 2021.
34. Bojian Yin, Federico Corradi, and Sander M. Bohté. Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. *Nature Machine Intelligence*, 3(10):905–913, Oct 2021.

Increasing the Location-Estimation accuracy of a Mobile Robot using a pre-trained Convolution Neural Network and a Depth-RGB-Camera

Niklas Förster

Department of Computing Science
Umeå University, Sweden
mrc22nfr@cs.umu.se

Abstract. Localisation of indoor robots is a major component in the robotic field. Nowadays, mostly odometry is used to estimate a robot's location, which is error-prone, specifically over long periods. An alternative is the use of an external device like a depth camera that estimates the absolute position of the robot and merges it with the relative position of the odometry to archive long-term position accuracy. Hence, this paper suggests using a depth-based camera in combination with a convolutional neural network (CNN) to accurately estimate the robot's position. The results suggest that fusing the found absolute position from the depth camera with odometry increases long-term position accuracy significantly, compared to odometry.

1 Introduction

Using indoor mobile robotics has become increasingly important in industrial and household applications. Unmanned Ground Vehicles, for logistics, are an example in industry, and autonomous vacuums in household applications. While there are solutions for the localization of mobile robotics available, most of them require an expensive hardware setup and calibration to be accurate or only use odometry which is error-prone and inaccurate, particularly for long-time applications. An option for small-scale applications could be the use of cheap camera systems, that allow the mobile robot to be localised. Particularly in environments where cameras are already common, like in restaurants for delivery robots, this option provides a cheap solution and could be implemented together with the surveillance system. This option can be realized with a single depth-RGB camera like a Microsoft Kinect v2. By fusing wheel odometry and the estimated position long-term accuracy of the position estimation can be achieved. This work will answer the question of what the impact on the estimated trajectory accuracy of a robot is when comparing the sole utilization of odometry against the proposed odometry-kinect solution. As a ground truth path, a polynomial will be defined and marked on the ground for the robot to follow. The comparison between ground truth and estimated trajectory will be done by calculating the minimal distance between the polynomial and each estimated position point.

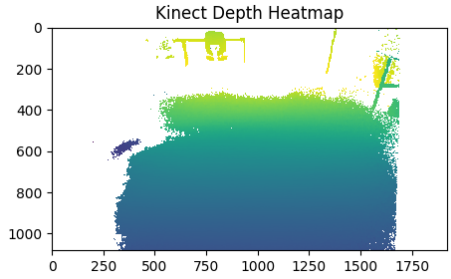
Suna Bensch (Eds.): Umeå's 27th Student Conference in Computing Science USCCS 2024, pp. 15–26, January 2024.

The experiments show a significant improvement using the Kinect-based system and odometry over long-term driving, compared to single odometry.

2 Background

Within this work, a dept-based RGB camera will be used. While there are many different companies producing depth cameras, Kinec v2 is easily accessible and compared to the Kinect v1, the standard deviation of the depth-data are not dependent on the measured distance itself and stay constant over the full measurement range ($0.5m$ to $4.5m$)[8]. With the Kinect v2, Microsoft provided a sensor, that is developed for gaming[9]. It measures the time of flight (ToF) of infrared beams and calculates the depth data accordingly. Additionally, the Kinect is equipped with an RGB sensor to take regular images and videos. The depth sensor has a resolution of 512x424 pixels, and the RGB-Sensor 1920x1080 pixels. The field of view of the Kinect is 70°H, 60°W for the depth data. The range of detection is between 0.5m to 4.5m. Figure 1 shows the type of images produced by a Kinect.

(a) RGB-Image of the test environment taken by Kinect v2 RGB-sensor with a resolution of 1920x1080 pixels and full field of view, where the robot is operating in.



(b) Depth-Image of test environment taken by Kinect v2 with colour graded depth-data, blue being closer, with fading into green for further distanced objects.

Fig. 1: The comparison of full resolution image in RGB and depth-data of Kinect in a test environment for the robot, with visual difference one field of view for the RGB-sensor and the depth sensor.

Calculating the odometry of a robot allows for an estimation of the position and the pose at all times. For that the x, y and θ position of the robot are continuously calculated by utilising sensors attached to the robot. In [4] the discrete-time equations used in this work are derived and explained. The final result is shown in equation 1, 2 and 3.

$$\theta(t + \Delta t) = \theta(t) + \frac{u_3(t)}{l}(\tan(\phi)\Delta t) \tag{1}$$

The angle θ is the orientation of the robot, also called pose, in a predefined absolute coordinate system, while ϕ represents the steering angles of the front wheels.

The variable $u_3(t)$ is the linear velocity in the middle of the robot's back axis, while l refers to the distance between the back wheel and the front wheel axis. Variable Δt is the discrete time interval between calculations/measurements.

$$x(t + \Delta t) = x(t) + \frac{u_3(t)}{\dot{\theta}}(\sin\theta(t + \Delta t) - \sin\theta t) \tag{2}$$

$$y(t + \Delta t) = y(t) + \frac{u_3(t)}{\dot{\theta}}(\sin\theta(t + \Delta t) - \sin\theta t) \tag{3}$$

With the pose θ, the coordinates x and y can be calculated. Calculating the odometry with these equations leads to the position and pose in reference to the starting position of the calculation. Because the Kinect uses its own predefined coordinate system and can be seen as absolute, a transformation between these two coordinate systems is needed. To go from one coordinate system to the other coordinate system with rotations and transformation, a homogeneous transformation matrix is used. The equations are defined and derived in [7]. The general homogenous transformation matrix can be written as the following:

$$H = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} \tag{4}$$

The rotation matrix $R = R_{XYZ}$ allows a 3-dimensional rotation of the coordinate frame. The entries of equation 5 are simplified and refer to $s = \sin$ and $c = cos$ with the angel being included in the footnote $c_\phi = \cos\phi$.

$$R = R_{XYZ} = \begin{bmatrix} c_\phi c_\theta - s_\phi c_\psi + c_\phi s_\theta s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi & -s_\theta c_\psi \\ s_\phi c_\theta c_\psi + s_\phi s_\theta s_\psi - c_\phi s_\psi + s_\phi s_\theta c_\psi & c_\phi c_\psi + s_\phi s_\theta s_\psi - c_\phi s_\psi + s_\phi s_\theta c_\psi & -s_\theta s_\psi \\ -s_\theta c_\theta s_\psi & -s_\theta c_\theta c_\psi & c_\theta \end{bmatrix} \tag{5}$$

The d vector in equation 4 is the translation from one to the other coordinate frame. It contains the x-y-z distance between the new and old systems. By multiplying a vector that is defined in the first coordinate frame with equation 4 the representation of the vector in the second coordinate frame is received, equation 6.

$$v_1 = Hv_0 = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} v_0 \tag{6}$$

3 Earlier Work

You Only Look Once (Yolo), implemented by [6], is an object detection Convolution Neural Network, that compared to other CNN and Computer vision deep learning algorithms processes an image with just one single iteration through the network. With only looking once at the image the processing detection time decreases, which makes it suitable for real-time applications that require object detection. Because of the limited CPU/GPU resources for this project yolov5

is a reasonable option for real-time visual object detection (robot-detection). In previous studies, the pre-trained yolov5 has been trained on custom data sets to enable smaller and more custom objects in a frame [5]. The author describes that real-time capability in general means between 20-30 frames per second of object detection, which will be a benchmark value for this work.

In the robotic community Kinect sensors are already utilised for various tasks. In [11], the author uses a Kinect v1 as a depth-RGB camera for detecting obstacles in front of the robot. The camera is attached on top of the robot and a computer is placed below it and connected to the camera. In [2], an additional neural network is used to get a robust collision detection system. However, this approach is not feasible because it requires an already precise odometry or localisation technique. In [3] the authors also used the Kinect attached to a mobile robot, however, they tested the camera's capabilities for mobile robotics in indoor and outdoor use. Because of the principle of time-off-flight measurement with infrared, direct sunlight disturbs the measurement. However, indoor use seems to deliver a consistent result, with higher inaccuracies at the edge of the image than in the middle. Because the best accuracy is archived in the middle of the image, in our work the robot is operated at the edge of the depth field of view to see if the worst case still provides enough precision. A system, similar to the one proposed in this paper, has been suggested by [10]. For that, a Kinect is used as an overhead camera in a robotic environment to detect obstacles, the pose and the location of multiple robots. In addition, IMU sensors and kinematic models of the robots are used to increase the precision of the estimation. Instead of using AI-based object detection a combination of the kinematic model and colour-graded parts on the robot are used to detect the robot and estimate the pose and position. From this information, the optimal position of the Kinect in our work will be determined.

4 Method

4.1 Training the Deep Neural Network Model

Using a Kinect sensor to return a distance value in the coordinates of a specific environment requires the knowledge of where the robot is in the field of view of the Kinect. For that purpose, the yolov5 pertained object detection model is used. To allow the CNN to detect a custom object, e.g. a robot, it is necessary to generate data that the model can be trained on. For this project, the Kinect will be placed in a fixed position, that allows for a wide field of view, so that the area of localization is maximized. Attaching the Kinect to the ceiling and facing downward would allow for easy coordinate transformation and a symmetric field of view, however, the ceiling also must be high enough and accessible. For this study, this was not possible, hence the Kinect is placed at a height of 1.5m with a tilt of 30° to face downwards (Figure 2). To reliably train a new object detection model, data have to be created that allow for teaching how an object looks. Therefore, within the test environment, the robot has to be placed in the field of view of the camera, and an image is taken. For training the CNN the image

Fig. 2: Placement of Kinect camera in Test-Environment on a shelf facing downwards toward the ground to allow for a maximized field of view.

has to be labeled and the robot specifically marked with a box. Each image that has been taken must be labelled like that. Figure 3 shows an example of that.

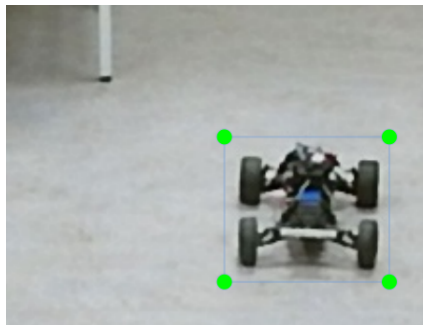


Fig. 3: Manual placed label example of object box around robot for creating the training and validation data-set for the pre-trained CNN.

The conflict between resolution and training/object-detection time has to be considered. While high resolution allows for a more detailed image of an object, it particularly increases the time for training. Particularly yolov5 is created for real-time applications and is optimised for images with the size 640x640 as training data. The training data for the custom yolov5 model are created by zooming into the original Kinect image and cropping it to 640x480 pixels, to keep the Kinect image aspect ratio. While the lab environment has only a few obstacles around, it is important to consider part of the area where obstacles are present. Otherwise, the CNN would show unstable behaviour for detecting the robot in areas where other objects are present as well. For training the object detection model up to 300 images are taken. An example of the custom-trained model and its capability of detecting a robot in the test environment is shown in Figure 4.

Fig. 4: Two examples (left and right) of the custom CNN model detecting a robot and labelling it with a red box and the certainty of the model that it actually is a robot ($0.8 = 80\%$).

4.2 Distance Calculation with Kinect

While Kinect provides an RGB image and a depth image at the same time, it uses different sensors with different resolutions for both data. However, the python-library libfreenect2 provides a method of merging both data and returns an RGB-Depth image. In the process of merging these images, the algorithm maps the depth data onto the RGB image and removes depth data or colour data that do not come with fitting data from each other. The CNN object detection model takes the original RGB image. After detecting a robot in the environment, the model returns indices for a rectangle around the robot in the image. With a predefined method from libfreenect2, the XYZ-Coordinate image from the Kinect perspective is calculated. The average depth data for the whole box in the coordinates with the same indices in the RGB-Depth-image are calculated and subsequently transposed into the environmental coordinate frame. Using the average allows for a more robust position estimation, than using only the middle of the rectangular. After calculating the position, it is sent to the robot and merged by setting the estimated position of the odometry to the new position. Hence, all positions of robots found by the Kinect will be accepted by the robot and merged without considering any delay.

4.3 Odoemtry and Fusion

For calculating the odometry the equations 1, 2 and 3 are used. Because the steering is done by a servo motor the steering angle angle of the tires is always known. Hence, the pose can be calculated each time the control program loops with a known time interval. The velocity is calculated by measuring the encoder data, which can be translated into the distance travelled in the known time interval. By dividing the distance by the time interval the velocity is calculated. As a base for the pose estimation of the robot, the odometry is taken, and the proposed method for this work is merging the found position of the robot provided by the Kinect without any filter or delay considerations into the odometry position.

4.4 Experimental Design

To allow the comparison between odometry, Kinect and the fused position, all of these position estimations are monitored and tested independently. However, to avoid deviations because of test differences, the tests are done simultaneously, while the robot is remote-controlled. All data are published to a computer and saved for analysis. To get a precise reference frame, markers will be placed on the floor, that are measured by hand. In that way, the coordinate frame for our system is defined. Because the Kinect system provides an absolute measurement in our coordinate frame it has to be calibrated. Hence the coordinate transformation between the Kinect coordinate frame and the origin of our environment will be checked by finding the position data of our markers in the Kinect. The odometry is a relative measurement, to allow for a consistent test a beginning marker will be placed, that allows for consistent beginning placement of the robot. As seen in Figure 5 the general experiment is to follow a line that can be

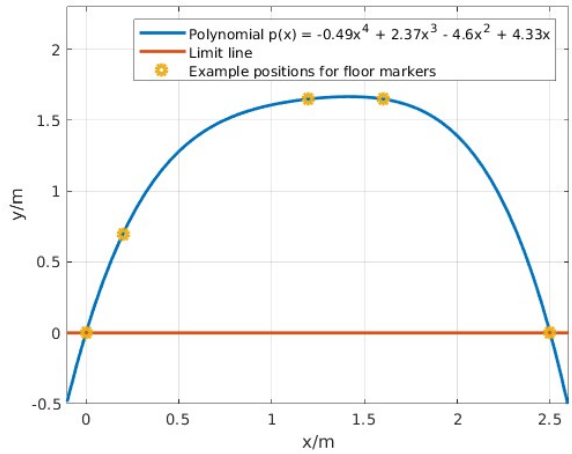


Fig. 5: Visualisation of the chosen optimal path for the robot, that is used as the ground truth for comparison with the estimated position of the robot, defined by a polynomial $p(x) = -0.49x^4 + 2.37x^3 - 4.6x^2 + 4.33x$. The example markers suggest positions that are taken from the polynomial and marked on the floor, the limit line shows the line below the robot position has not been recorded.

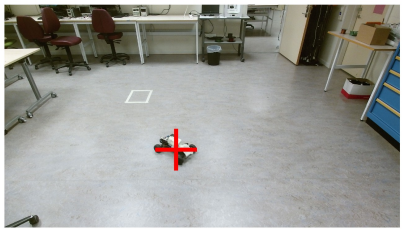
represented by a specific polynomial $p(x) = -0.49x^4 + 2.37x^3 - 4.6x^2 + 4.33x$. To allow the robot to follow the curve precisely, the coordinates of the polynomial are marked every 20cm on the floor. The lower part of the robot trajectory below $x = 0$ is not considered for the test. The upper part of the robot trajectory will be used to calculate the minimal distance to the optimal path.

5 Results

The relevant results of this work can be divided into two parts. The first part is the object detection robustness of the computer vision CNN model. The second part is the localization of the robot with the depth sensor of the Kinect.

5.1 Robot Detection Robustness

Within the test environment, the object detection of the robot works 74% of the time correctly while driving in a circular curve. For the remaining 26% the robot is not detected at all. An example of the detection circumstances is shown



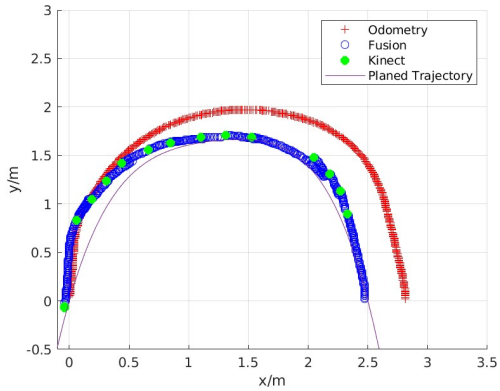
(a) The detected robot, marked with a red cross, while it faces forward towards the camera, using the CNN model and the Kinect-RGB-image in full scale.

(b) Undetected robot which is suggested by no red cross, using the CNN model and the full-scale Kinect-RGB-image. (robot in the right corner)

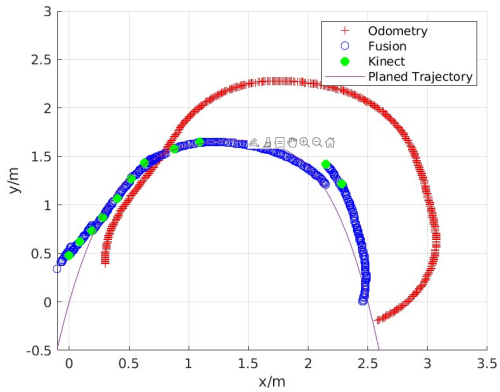
in Figure 6b and Figure 6a. It is important to assess why the detection works in some cases and in others, it does not. Figure 6a shows the detection of the robot, which happens mostly when the robot approaches the Kinect sensor. When moving away from the sensor the object detection seems to lose its ability to recognize the robot, as seen in Figure 6b. One reason for this can be the trained CNN model. Even when considering that the model has been pre-trained, only 300 images have been used to train the neural network for detecting the new robot. With using more images the detection of the robot might become more accurate when it faces away from the camera. However, with no false positives in the test scenario, the merging between odometry and Kinect position still works well and shows a significant improvement in the estimated position of the robot.

5.2 Localisation of Robot

In Figure 7 the difference between the planned trajectory, the fused position, and the odometry of the robot are shown. Figure 7a the deviation between fused position and odometry is already visible. When considering Figure 7b the odometry trajectory continues to wander off from the optimal trajectory further to the right. The fused trajectory, however, shows no obvious deviation compared to the optimal. With the previous result from the object detection robustness, it is also apparent here, that in some cases the robot seems to be detected



(a) The estimated odometry and fused odometry-Kinect trajectory in reference to the optimal path-polynomial in the first round of the robot driving.



(b) The estimated odometry and fused odometry-Kinect trajectory in reference to the optimal path-polynomial in the fifth round of the robot driving

Fig. 7: Comparision between estimated odometry, odometry-Kinect trajectory and the optimal path-polynomial for the first round of operating the robot and for the fifth round of operating.

more regularly than in others. Close to $x = 0$ and $y = 0$ the detection seems to work regularly, with positions close to $x = 2.5$ the detection happens less regularly. When looking at the mean minimal distance for each point from the

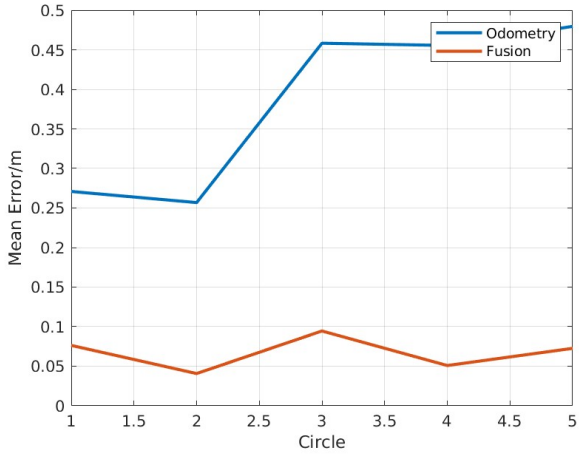


Fig. 8: Mean minimal distance of odometry trajectory, odometry-Kinect trajectory to the ideal polynomial path of the robot, plotted for each circle the robot was operating in the environment.

odometry trajectory and the fused odometry-kinect algorithm compared to the optimal path, the previous observations, that the fused trajectory stays closer to the optimal, are confirmed. Figure 8 shows the mean minimal distance for both position estimation algorithms, with each loop shown individually. For the odometry the minimal distance increases by each loop around the track. The fuse position estimation algorithm, on the other hand, shows a consistent and similar result in each loop. The same behaviour appears when the standard deviation of the minimal distance is plotted. The odometry shows an increase by the amount of loops done by the robot, while the fused algorithm appears to stay the same while continuing to loop through the test environment. The plot is shown in Figure 9.

6 Conclusion

While using more images to train the model is one option, it also should be considered to use a Kalman-Filter or a similar sensor merging algorithm that prevents false data from being accepted into the robot position.

Another aspect to consider here is the real-time detection of the robot. The computer on which the tests are run made it possible to run the algorithm to detect the robot, however, there was no GPU installed. The consequence is a slower detection rate than other systems would provide with GPU. In general,

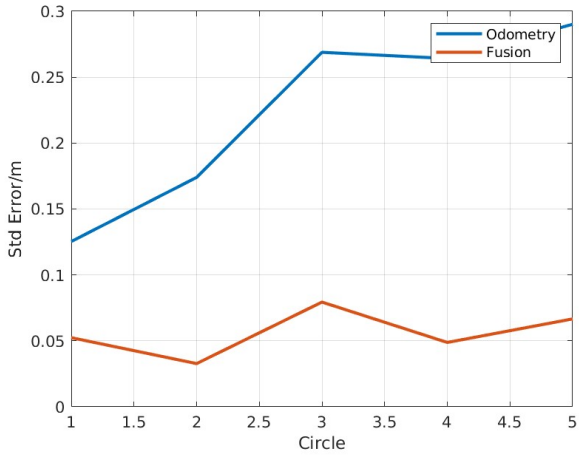


Fig. 9: Standard deviation of the minimal distance of the odometry trajectory, odometry-Kinect trajectory to the ideal polynomial path of the robot, plotted for each circle the robot was operating in the environment.

using a neural network, GPUs provide a better base system than CPUs, hence, they are preferred in computer vision and AI applications. [1].

Additionally, the results shown in this paper only provide a 2-dimensional consideration of the deviation between the optimal path and the recorded path, however, the time aspect has not been considered. Especially with the duration of 200ms of detection time of the robot, depending on the speed, the estimated new position might be outdated already. To consider partially this aspect in this work, the robot is controlled slowly with around $20\frac{cm}{s}$. This would lead to around $4cm$ of delay distance from taking an image and merging the location into the robot.

As described before, the positioning of the Kinect-Sensor is not optimal. Because the rectangular of the detected object is used for finding the middle of the object, sometimes the middle of a side-down view leads to using the side of the robot as the middle. Hence, the accuracy decreases. For future work, it is suggested, to implement an algorithm, that detects the robot and then further analyses where the middle of it is specifically. To decrease the complexity of this, it is beneficial to place the Kinect sensor at the ceiling and have it face downwards.

One aspect to consider is the privacy concern. Because this project proposes a localisation of a robot via camera, it is limited in the areas where it can be used. In household applications, it is unlikely to succeed. However, areas where cameras are already used can consider this kind of solution.

In conclusion, the Kinect-Odometry fusion shows significant improvement towards the used odometry in this test. Even with a single Kinect camera, it can provide a reliable solution for updating the absolute position of a robot after a while. With the limited field of view of the camera, the robot has to pass through

the camera view regularly. Depending on the accuracy of the odometry that has to happen more or less often.

References

1. Ebubekir Buber and Banu Diri. Performance analysis and cpu vs gpu comparison for deep learning. In *2018 6th International Conference on Control Engineering & Information Technology (CEIT)*, pages 1–6, 10 2018.
2. Diogo Santos Ortiz Correa, Diego Fernando Sciotti, Marcos Gomes Prado, Daniel Oliva Sales, Denis Fernando Wolf, and Fernando Santos Osorio. Mobile robots navigation in indoor environments using kinect sensor. In *2012 Second Brazilian Conference on Critical Embedded Systems*, pages 36–41, 2012.
3. Péter Fankhauser, Michael Bloesch, Diego Rodriguez, Ralf Kaestner, Marco Hutter, and Roland Siegwart. Kinect v2 for mobile robot navigation: Evaluation and modeling. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 388–394, 2015.
4. Yi Fu, Howard Li, and Mary Kaye. Design and stability analysis of a fuzzy controller for autonomous road following. In *2009 IEEE Intelligent Vehicles Symposium*, pages 66 – 71, 07 2009.
5. Mohammad Hossein Hamzenejadi and Hadis Mohseni. Fine-tuned yolov5 for real-time vehicle detection in uav imagery: Architectural improvements and performance boost. *Expert Systems with Applications*, 231:120845, 2023.
6. Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 06 2016.
7. Mark Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. John Wiley & Sons, 2020.
8. Michal Tölgyessy, Martin Dekan, and Lubos Chovanec. Skeleton tracking accuracy and precision evaluation of kinect v1, kinect v2, and the azure kinect. *Applied Sciences*, 11(12), 2021.
9. Michal Tölgyessy, Martin Dekan, LuboS Chovanec, and Peter Hubinsky. Evaluation of the azure kinect and its comparison to kinect v1 and kinect v2. *Journal of Sensors*, 2021(2), 2021.
10. Doopalam Tuvshinjargal, Byambaa Dorj, and Deokjin Lee. Hybrid motion planning method for autonomous robots using kinect based sensor fusion and virtual plane approach in dynamic environments. *Journal of Sensors*, 2015:1–13, 07 2015.
11. Nor Zainuddin, Yasir Mohd Mustafah, Shawgi Mounis, and Nahrul Khair Alang Md Rashid. Autonomous navigation of mobile robot using kinect sensor. In *2014 International Conference on Computer and Communication Engineering*, pages 28–31, 2014.

Path Planning Algorithm for UAV in 3D Space Based on the Fusion of A* and RRT

Zeyu Gao

Department of Computing Science
Umeå University, Sweden
zega0002@student.umu.se

Abstract. This paper introduces a fusion algorithm for UAV (Unmanned Aerial Vehicle) path planning in 3D space. It combines the A* algorithm and the RRT (Rapidly Exploring Random Tree) algorithm, two traditional path planning methods, and verifies its performance through simulation experiments. The fusion algorithm first searches the nodes rapidly by the RRT algorithm and then activates the A* algorithm under specific conditions to optimize the path to the target point. The results of the simulation experiments demonstrate that the fusion algorithm consumes significantly less time and generates fewer nodes than the two traditional algorithms do separately, and also saves 54% and 26% of CPU usage respectively.

1 Introduction

With the rapid advancement of technology, Unmanned Aerial Vehicles (UAVs) have found extensive applications across various domains, including military, agriculture, research, education, entertainment, and logistics. Autonomous flight of UAVs, which enables drones to operate independently without direct human control, is a crucial technology that has garnered extensive research worldwide. This field is primarily divided into four key areas [5]: perception, localization, motion control, and path planning, with path planning being the most critical among them. Path planning is the process of determining the flight path for UAV, considering obstacles, efficiency, and specific mission goals. Currently, path planning algorithms can be broadly categorized into traditional algorithms and intelligent algorithms. Among the traditional algorithms, A* [4] and Rapidly Exploring Random Tree (RRT) [7] are quite popular. These two traditional algorithms and their variants exhibit their respective strengths in solving 3D path planning problems for UAV. The A* algorithm consistently generates optimal paths, while the RRT algorithm excels in terms of speed in path generation. Leveraging the advantages of both algorithms, a fusion algorithm combining their strengths holds significant potential for addressing complex UAV path planning challenges. Therefore, this paper proposes a fusion algorithm based on the A* algorithm and the RRT algorithm, which rapidly expands the nodes to approach the goal area by the RRT algorithm and then activates the A* algorithm to optimize the last segment of the path. Simulation experiments are designed and run to verify the advantage of the fusion algorithm.

Suna Bensch (Eds.): Umeå's 27th Student Conference in Computing Science USCCS 2024, pp. 27–37, January 2024.

2 Background

This section describes how the A* algorithm and the RRT algorithm work in principles, respectively. It also helps to understand how the fusion algorithm works in the later section.

2.1 The A* Algorithm

The A* search algorithm [4] is a heuristic search algorithm used to find the path with the lowest cost through multiple nodes in a graph or grid. It can find the shortest path. The core of the A* algorithm lies in the design of its evaluation function, which is represented as:

$$f(n) = g(n) + h(n) \tag{1}$$

- $f(n)$ represents the estimated cost for each possible probing point.
- $g(n)$ represents the cost from the initial search point to the current point.
- $h(n)$ represents the heuristic estimate cost from the current node to the goal.

From this, it can be seen that the evaluation function evaluates nodes by combining the cost to reach the node and the cost to get from the node to the goal.

The algorithm initiates by setting the costs of the starting node, which serves as the initial search point, and then places it in an open list. It iteratively selects nodes with the lowest cost from the open list, moving them to a closed list. If the goal node is reached, the path is found. If not, neighboring nodes are explored. The algorithm updates costs and keeps track of the parent nodes. If the open list becomes empty without reaching the goal, the algorithm concludes that there is no solution. A* algorithm's time complexity is not fixed, it highly depends on the heuristic used, the graph's structure, and the implementation details. Algorithm 1 describes the underlying principles of the A* Algorithm.

2.2 RRT Algorithm

The Rapidly Exploring Random Tree (RRT) algorithm [7] is a sampling-based search algorithm used in high-dimensional space exploration. This algorithm generates a tree with the initial configuration, which is the starting point as the root by using random samples from the search space. The search space is the domain within which the RRT operates to find a path. It is the coordinates in a continuous three-dimensional space and also includes constraints that the algorithm must bypass, such as obstacles. When each sample is drawn, an attempt is made to establish a connection between it and the nearest state in the tree. If the connection is feasible (completely within free space and compliant with any constraints), the new state is added to the tree. By increasing the probability of sampling states from specific regions, RRT can introduce bias in its growth, guiding the tree towards the goal. The time complexity of RRT depends on factors like the dimensionality of the space, the distribution of obstacles. In general, it can be considered polynomial with respect to the number of nodes in the tree. Algorithm 2 describes the underlying principles of the RRT Algorithm.

Algorithm 1 A* Algorithm

Add *initN* to *openList*
$g(init)$=0, $h(init)$=$h(start, goal)$, F=$h(init)$ with *parent* as **null**
while *openList* is not empty **do**
    Retrieve N with the lowest F as *curN* from the *openList*
    Add *curN* to the *closedList*
    if *curN*=*goalN* **then**
        Path found
        return the path from start to *goalN*
    else
        for each $Neighbor$ **do**
            if $Neighbor$ is in *closedList* **then**
                Skip this N
            else if $Neighbor$ is not in *openList* **then**
                Add $Neighbor$ to *openList*, set *parent* to *curN*
                Calculate *g(n)*, *h(n)*, *f(n)*
            else if $Neighbor$ is in *openList* **then**
                Calculate the new *g(n)*
                if the new *g(n)* <*g(n)* **then**
                    Update *parent* to *curN* and recompute *g(n)*, *h(n)*, *f(n)*
                end if
            end if
        end for
    end if
end while
if *openList* is empty and endpoint not found **then**
    No solution exists
    return null
end if

Algorithm 2 RRT Algorithm

Input: Initial configuration q_{init}, number of vertices in RRT K, incremental distance Δq
Output: RRT graph G
$G.init(q_{init})$
for $k = 1$ to K **do**
    $q_{rand} \leftarrow$ random configuration within the search space
    $q_{near} \leftarrow$ closest vertex in G to q_{rand}
    $q_{new} \leftarrow$ point towards q_{rand} from q_{near} within Δq
    $G.add\_vertex(q_{new})$
    $G.add\_edge(q_{near}, q_{new})$
end for
Return G

3 Earlier Work

Currently, numerous variants of both A* algorithm and RRT algorithm have been developed. Most of these variants aim to reduce the consumption of computational resources by the traditional algorithms or to optimize the paths.

The Dynamic A* algorithm (D*) [8] is an evolution of the A* algorithm. This algorithm effectively addresses the issue of changing cost parameters during the search for a solution by only updating new nodes. Consequently, it can efficiently adapt to dynamic environments.

Another algorithm that is based on interpolation planning and re-planning is called 'The field D*' [3]. This method uses linear interpolation during the planning process to compute precise path cost estimates for any position within each grid cell. It generates paths with a series of continuous directions, thereby reducing computation time and lowering memory requirements during computations.

Characterized as an incremental heuristic algorithm, the Generalized Adaptive A* [9] enhances its function using knowledge from past searches. This algorithm uses knowledge obtained from previous searches to update its heuristic function, thereby improving efficiency.

Both the Fringe-Saving A* [10] algorithm and the Differential A* [11] algorithm reduce unnecessary re-computation and improve efficiency by identifying regions that have remained unchanged since the previous calculation and continuing the search from there.

The RRT-Connect algorithm [6] is a variant of the RRT algorithm which simultaneously constructs the tree toward the goal node. Furthermore, the Execution-Extended RRT (ERRT) algorithm [2] tackles path failures and introduces probabilistic re-planning in dynamic environments. Path failures occur when changes render the current path unfeasible or suboptimal, while probabilistic re-planning involves recalculating paths due to these changes. ERRT enhances this by using new random samples and nodes from prior unsuccessful paths.

Incorporating Ant Colony Optimization (ACO) [12] for optimal path planning, the method draws inspiration from the natural movement of ant colonies. This approach simulates the movement of ant colonies in nature, where pheromone information is placed on the paths obtained by the RRT. The next expansion point is selected based on the concentration of pheromones. Through a certain number of iterations, the algorithm converges to an ideal path solution.

A combination of RRT and A* to solve path planning problems in 2D space can be found in [1]. It uses A* to optimize the nodes generated by RRT. As a result, the fusion algorithm significantly reduces the number of generated nodes, decreasing computational costs and saving time, although the limitations of 2D space result in the algorithm not being scalable to 3D space.

The various variants of A* and RRT algorithms in different fields demonstrate that both of the two are mature and advanced path planning algorithms. The application of fusion algorithms in 2D space has also shown their superiority compared to traditional algorithms.

A comparison of the A*, RRT, and their variants, which were introduced above, in 3D UAV path planning environments, can be found in [13]. The A* algorithm consistently generates optimal paths, while the RRT algorithm tends to produce non-optimal paths. Moreover, A* typically has lower computational requirements compared to RRT. As a result, A* performs better in 3D path planning applications with static or dynamic obstacles. Additionally, A* allows for different discretizations of the environment based on varying levels of urgency in different parts of the scene, optimizing resource utilization. However, if random sampling of nodes is done intelligently based on obstacle shapes, positions, RRT and its variants can outperform A* in terms of path construction time. The above comparison outlines the strengths of two traditional algorithms in 3D UAV path planning problems, demonstrating the feasibility of fusion algorithms in 3D space.

4 Method

4.1 Fusion Algorithm

The fusion algorithm is divided into two phases. The first phase involves running the RRT algorithm to rapidly generate nodes, while the second phase uses the A* algorithm to optimize the nodes generated by the RRT algorithm. There exists a determination between the two phases to determine the timing to proceed to the second phase, which is the key to the fusion algorithm. Figure 1 demonstrates the flow chart of the fusion algorithm. The following is a detailed introduction to the entire process of the fusion algorithm.

In the first phase of the algorithm, the RRT algorithm rapidly extends the exploration range by starting from an initial point, randomly generating new sample points, and adding the new points to the tree structure subject to specific step size and obstacle avoidance conditions. If the algorithm finds the path at this phase it returns the path directly without proceeding to the next phase. When a new node reaches the predefined radius of the goal area, it triggers the condition to proceed to the next stage.

In the second phase, the algorithm needs to first determine whether the distance between the new node and the last searched node exceeds the preset value, if it is less than the preset value then it returns to the first phase, otherwise it activates the A* algorithm. The A* algorithm is activated to minutely search the localized area to find the goal point.

4.2 Simulation Environment and Set up

Simulation experiments are conducted to verify the feasibility of the fusion algorithm and to evaluate the performance of the algorithm. The experiment uses Matlab R2020a running on Macbook Air M1, 64-bits MacOS operating system, Apple M1 CPU, 8GB unified memory.

To construct the map for the simulation experiment, a $170 \times 170 \times 170$ space is created with randomly placed and differently sized spheres serving as obstacles.

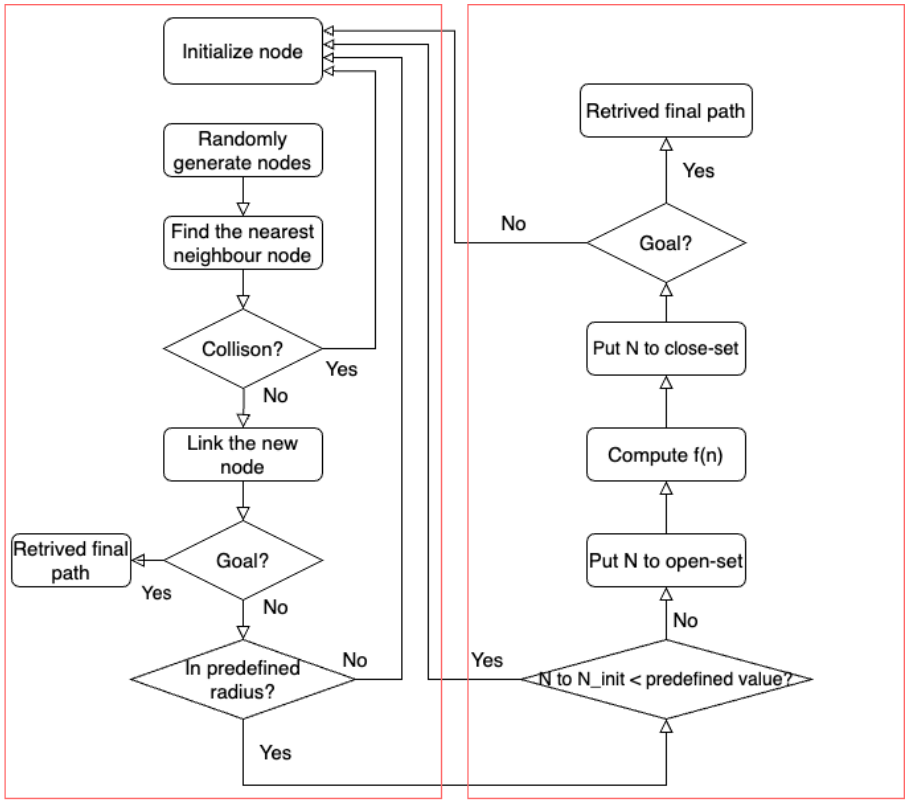


Fig. 1. The flow chart of the Fusion Algorithm includes Phase 1 (Left red rectangle) and Phase 2 (Right red rectangle).

The green and blue points are used to indicate the starting point and goal point respectively. In this way three maps (Map-1, 2, 3) were generated to be used as three sets of experiments (Experiment-1, 2, 3). Figure 2 containing the three maps shows the number, size, and location of the obstacles in the space, and also indicates the location of the start and goal points.

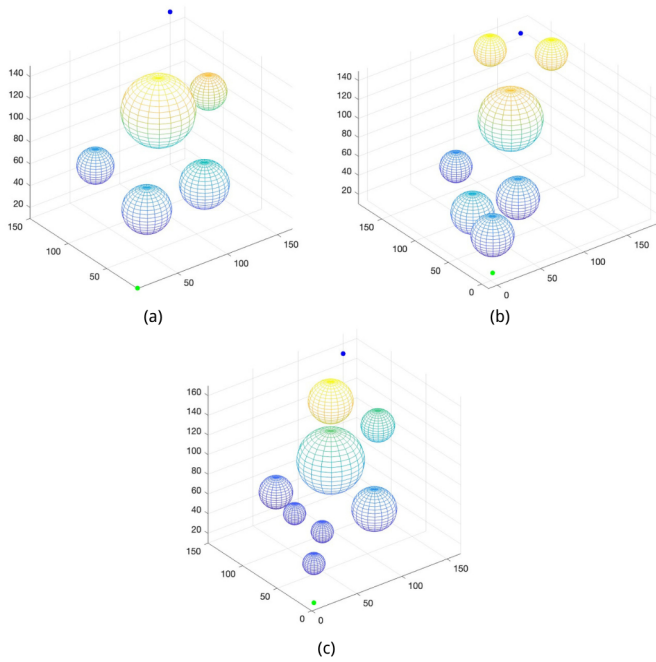


Fig. 2. Maps for experiments Experiment-1 (a), Experiment-2 (b), and Experiment-3 (c) include the starting point (green), the goal point (blue), and obstacles (spheres).

Before starting the experiment, some parameters need to be initialized and kept the same in each set of experiments. The important parameters are as follows:

- Start point: $(10, 10, 10)$
- Goal point: $(150, 150, 150)$
- A* threshold: 25 (density of nodes)
- Goal threshold: 37.5
- Step size: 10
- Max failed attempts: 10000

4.3 Experiments

The experiments are conducted in three (Experiment-1, 2, 3) sets, using one map (as shown in Figure 2) for each set of experiments. In the experiment, each of

the three algorithms (A*, RRT, and FusionAlgorithm) performs a path search beginning from the starting point until a path to the goal point is found and drawn. The time consumed by each algorithm, the number of nodes unfolded and the length of the final path generated are recorded during the process, meanwhile the CPU usage of each algorithm is monitored during its operation. The recorded data is used in the next section to analyze and compare the performance of the algorithms.

5 Results

The paths generated by the three algorithms in Experiment-1 are shown in Figure 3, where the colored lines and dots represent the generated nodes and paths, and the red lines indicate the final paths searched. It is clear from the figure that the fusion algorithm unfolds fewer branches. The complete results of the experiments are demonstrated in Table 1, and Figure 4 is a bar chart generated for comparison purposes.

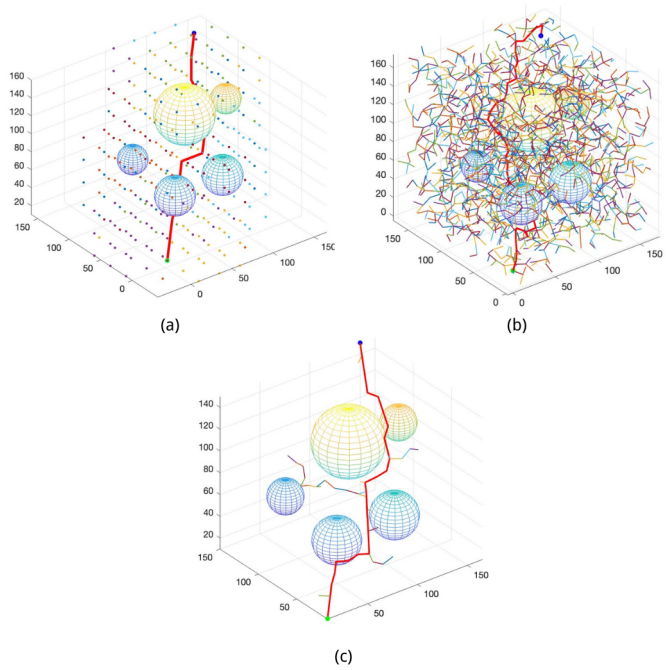


Fig. 3. The paths generated by the three algorithms in Experiment-1: (a) A* (b) RRT (c) FusionAlgorithm.

Set	Algorithm	Time	Nodes	Path
Map 1	A*	14.45s	3191	276.59
	RRT	129.31s	7061	398.97
	FusionAlgorithm	3.34s	97	303.24
Map 2	A*	12.08s	2884	276.59
	RRT	76.85s	3124	402.64
	FusionAlgorithm	4.92s	210	290.24
Map 3	A*	16.07s	3442	276.59
	RRT	87.01s	3681	404.32
	FusionAlgorithm	3.19s	116	318.48

Table 1. Results for time consumed, nodes generated and path length by different algorithms across various set of experiments.

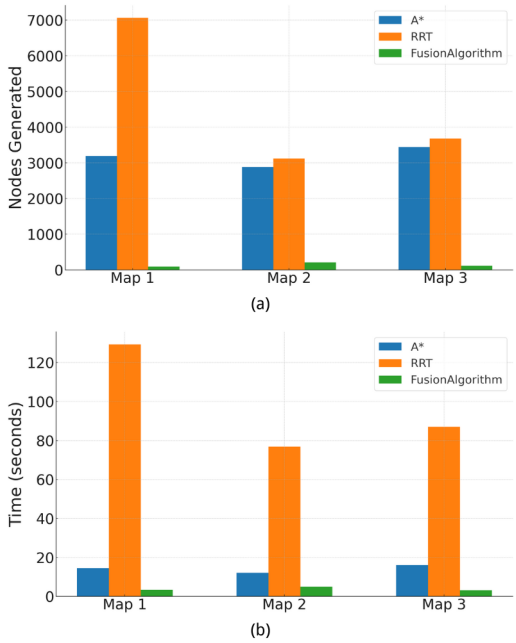


Fig. 4. Comparative results of three algorithms. (a) Nodes generated. (b) Time consumed.

Algorithm	CPU
A*	112%
RRT	83%
FusionAlgorithm	57%

Table 2. The average CPU usage during the running of the three algorithms.

In all the three sets of experiments, the fusion algorithm shows an average decrease of about 73.12% in time and about 95.56% in the number of nodes generated as compared to the A* algorithm. When compared to the RRT algorithm, the fusion algorithm shows an average decrease of about 96.09% in time and about 96.95% in the number of nodes generated. The paths of the fusion algorithm are on average 24.38% shorter compared to the length of the paths generated by the RRT. In principle, when a node reaches within a predetermined radius of the target point, the second stage is activated to perform a more minutely search instead of rapidly unfolding more nodes as in the first stage, which is why the fusion algorithm significantly reduces the number of nodes. However, since the fusion algorithm is based on the RRT algorithm and cannot generate optimal paths, the paths are still longer than those generated by the A* algorithm. The average CPU usage[1] during the running of the three algorithms is shown in Table 2, with the fusion algorithm using the least amount of CPU. Generating new nodes takes up Random Access Memory, and fewer nodes represent less memory usage. These two facts demonstrate that the fusion algorithm can significantly save computational resources.

6 Conclusions

In this paper, a fusion algorithm based on the A* algorithm and the RRT algorithm is proposed. The fusion algorithm first uses the RRT algorithm to quickly generate nodes during operation, and then activates the A* algorithm to optimize the paths in the region after reaching the vicinity of the goal point. The results of the experiment demonstrate that the fusion algorithm significantly reduces the time and computational resources consumed for path planning. Additionally the length of the path generated by the fusion algorithm is also noticeably shortened compared to the Basic RRT algorithm. These advantages make it an effective solution to the 3D space UAV path planning problem.

The main aim of the future research is to solve the limitation that the fusion algorithm is unable to search the optimal path due to the nature of the RRT-based algorithm, and to optimize the path by introducing a smoothing algorithm. Meanwhile, the use of fusion algorithms to quickly generate time-varying paths in complex dynamic spaces should also be a key focus of future research.

References

1. Suhaib Al-Ansarry and Salah Al-Darraji. Hybrid RRT-A: An improved path planning method for an autonomous mobile robots. *Iraqi journal for electrical and electronic engineering*, 17(1):1–9, 2021.
2. James Bruce and Manuela M. Veloso. Real-time randomized path planning for robot navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2383–2388, 2002.

[1] In MacOS, the CPU usage is calculated per core, so with multi-core processors, the total can surpass 100% as each core's full capacity counts as 100%.

3. Dave Ferguson and Anthony Stentz. Using interpolation to improve path planning: The field D* algorithm. *Journal of Field Robotics*, 23(2):79–101, 2006.
4. Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on systems science and cybernetics*, 4(2):100–107, 2022.
5. Anantha Injarapu and Suresh K. Gawre. A survey of autonomous mobile robot path planning approaches. In *International Conference on Recent Innovations in Signal Processing and Embedded Systems (RISE)*, pages 624–628, 2017.
6. James Kuffner and Steven M. LaValle. RRT-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation*, pages 995–1001, 2000.
7. Steven M. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report TR 98-11, Dept. of Computing Science, Iowa State University, America, 1998.
8. Anthony Stentz. Optimal and efficient path planning for unknown and dynamic environments. *International journal of robotics and automation*, 10(3):89–100, 1995.
9. Xiaoxun Sun, William Yeoh, and Sven Koenig. Generalized adaptive A*. In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent systems (AAMAS)*, pages 469–476, 2008.
10. Xiaoxun Sun, William Yeoh, and Sven Koenig. Dynamic fringe-saving A*. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent systems (AAMAS)*, pages 891–898, 2009.
11. Karen Trovato and Leo Dorst. Differential A*. *IEEE transactions on knowledge and data engineering*, 14(6):1218–1229, 2002.
12. Fan Yang, Xi Fang, Fei Gao, Xianjin Zhou, Hao Li, Hongbin Jin, Yu Song, Shi Cheng, and Shi Cheng. Obstacle avoidance path planning for UAV based on improved RRT algorithm. *Discrete dynamics in nature and society*, 2022(1):1–9, 2022.
13. Christian Zammit and Erik-Jan van Kampen. Comparison between A and RRT algorithms for 3D UAV path planning. *Unmanned systems (Singapore)*, 10(2):129–146, 2022.

Group Theory of the 3×3×1-Rubik's Cube

Pina Kolling

Department of Computing Science
Umeå University, Sweden
ens21pkg@cs.umu.se

Abstract. The 3×3×3-Rubik's Cube is a popular puzzle that is traditionally analysed using group theory to understand its complex permutations and visualise the algebraic concept of groups. In this paper, the 3×3×1-Rubik's Cube is represented with group theory to contribute to the expanding scope of mathematics. Using the principles of quotient sets, equivalence classes and permutation functions, a representation in form of the algebraic structure of a group that captures the cube's characteristics is created. For this, the configuration of the cube is defined as a 2-tuple, representing the position of the corner pieces and the orientation of the edge pieces. This group theoretic model is then used to analyse the cube: The amount of possible configurations is determined by determining the order of the group, and trees are modelled and compared as methods to find the optimal solution for solving the cube.

1 Introduction

The 3×3×3-Rubik's Cube is a mathematical rotating puzzle invented in 1974 by the Hungarian professor Ernő Rubik that has been the subject of numerous papers: It has been presented and studied as a group in [1] and [6] to visualise the complex topic of group theory and algebra and to get further knowledge about the 3×3×3-Cube. [2] focusses on the teaching of group theory and uses the group theoretic model of the 3×3×3-Cube as a visualization to illustrate abstract mathematical concepts such as subgroups, homomorphisms, and equivalence relations. The *God's Number* indicates the maximum number of moves needed to solve the cube when using the optimal solution path. Based on the group theoretical model the God's Number has been attempted to compute in [9] and been calculated and proven in [10]. Furthermore, different solving methods for humans have been compared in [4] and finding an optimal solution part has been examined in [3]. In [7] the group theory from the 3×3×3-Cube has been transferred to the 2×2×2-Cube and it has been examined on mechanical solving methods with image and colour recognition in [8].

In 2004 the 3×3×1-Cube (so called *Floppy Cube* or *Rubik's Edge*) was invented by the Japanese inventor Katsuhiko Okamoto, who specialises on Rubik's

Suna Bensch (Eds.): Umeå's 27th Student Conference in Computing Science USCCS 2024, pp. 39–55, January 2024.

Cube modifications[1]. The 3×3×1-Cube consists of a single layer with 9 smaller cubes (see Figure 2) while the 3×3×3-Cube has three layers (Figure 1).

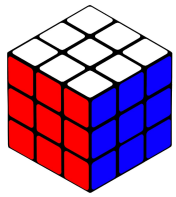


Fig. 1. A solved 3×3×3-Cube that consists of 26 smaller cubes.

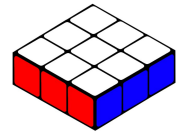


Fig. 2. A solved 3×3×1-Cube that consists of 9 smaller cubes.

The aim of this paper is to extend the knowledge of the 3×3×1-Cube. Exploring different mechanical contraptions (for example Rubik's Cubes) with the help of different algebraic concepts including group theory expands the mathematical knowledge base of humanity and is creating new opportunities and insights for solving new problems or examining other mathematical or algorithmical problems. Mathematical research, even if not immediately applied, expands the foundation of knowledge, leading to unforeseen applications and insights that are essential for the continuous development of the field.

In contrast to the 3×3×3-Cube, the 3×3×1-Cube has fewer individual pieces (also referred to as *cubies*, as described in Section 2) and only one level – this makes the smaller cube less complex as it has less possible configurations. On the other hand, the 3×3×1-Cube can shape shift, which refers to the cube's ability to change its overall form as a result of turning its layers (Figure 6). This brings up limitations on the design of the group of the 3×3×1-Cube compared to the group of the 3×3×3-Cube, including permutation functions that do not only permute cubies but also empty spots. The limitations are further described in Section 4.

In this paper, the group theory of the 3×3×1-Cube is defined. With the design of the 3×3×1-Cube mechanics as a group, knowledge about the cube can be acquired. This includes calculating the number of possible cube configurations and the creation of a concept for finding the optimal solution path using a tree. While presenting a concept for identifying the optimal solution path, the optimal solution path is not determined in this paper and its exploration remains as future research.

This paper is divided into different sections: Section 2 explains the structure of the 3×3×1-Cube, the corresponding terminology and the basic moves. The mathematical basics including group theory are defined and explained in Section 3. In Section 4, the position and orientation of the cubies and the execution of moves of the cube are mathematically defined. Based on this, the equivalence classes of the moves are defined, leading to a quotient set of the possible moves

[1] `https://www.grubiks.com/puzzles/floppy-cube-1x3x3/` and `https://ruwix.com/twisty-puzzles/super-floppy-cube/`, Websites about puzzles, accessed 13.09.2023

of the cube. The group of the cube, using the previously defined quotient set, is described and proven in Section 5 by examining the four group axioms: Closure, associativity, existence of a neutral element and existence of inverse elements. Section 6 contains the order of the group and a tree concept for finding a minimal solution. Section 7 contains the conclusion, summary, results and future work.

2 The 3×3×1-Cube

In this chapter, the basic mechanics, structure and terminology of the 3×3×1-Cube and the four basic moves of the cube are defined and explained. The 3×3×1-Cube consists of 9 smaller cubes, that are referred to as *cubies*.

Corner cubie The 3×3×1-Cube has four corner cubies (Figure 4). Each of these corner cubies has four colour panels. Figure 3 displays one colour panel.

Edge cubie The 3×3×1-Cube has four edge cubies with five colour panels each (Figure 5). Two of the five colour panels are hidden when the cube is in the start configuration. The shape shifting properties of the 3×3×1-Cube expose more of the colour panels, depending on the configuration of the cube (Figure 6).

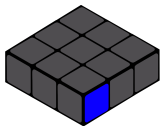


Fig. 3. One of the 30 colour panels is shown.

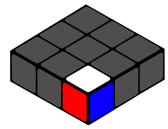


Fig. 4. One of the four corner cubies is marked.

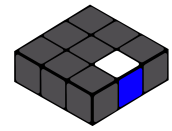


Fig. 5. One of the four edge cubies is marked.

3×3×1-Cube (also called *Floppy Cube*)
Figure 6 shows the cube on the left in its solved configuration and in the middle and right scrambled. The solved configuration is also referred to as start configuration. In its start configuration (also referred to as solved configuration), each side of the cube shows only colour panels of one colour. In a scrambled variation, the colours can be in different positions and the 3×3×1-Cube has the ability to change its overall shape when being scrambled, which means that it does not stay in its rectangular form (Figure 6).

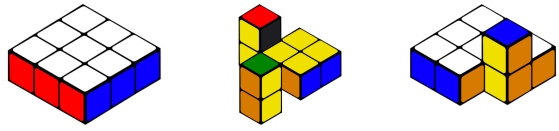


Fig. 6. A solved configuration of the 3×3×1-Cube (left) and two scrambled configurations with visible shape shifts (middle and right) are shown.

Side The 3×3×1-Cube has six sides. The white side is typically considered the top side in the start configuration and the colour pairs on the opposite sides of the cubes are white and yellow; red and orange; green and blue.

2.1 Moves of the Cube

The 3×3×1-Cube has four different turnable sides, because the middle cubie has a mechanism that allows turning on four of its six sides. The table below (Figure 7) contains the basic moves of the cube. The naming of the moves follows the name convention of the cubing community[2]. Each move is defined by turning a side clockwise when facing that side. The cube configurations in Figure Figure 6 be reached by the moves *LB* (configuration in the middle) and *LBLLL* (configuration on the right).

Name	Description of the move
R	Turning the **right** side clockwise
L	Turning the **left** side clockwise
F	Turning the **front** side clockwise
B	Turning the **back** side clockwise

Fig. 7. Basic moves of the 3×3×1-Cube.

3 Mathematical background

The mathematical definitions in this section are based on [5]. This are essential for gaining a comprehensive understanding of the cube and it starts with the definition of groups and other algebraic structures.

Definition 1 (Group). *A (non-empty) set G with a binary operation $\circ$ is called* group *(denoted as $(G, \circ)$), if the following* group axioms *are satisfied:*

- *Closure:* $\forall\, a, b \in G \,.\, (a \circ b) \in G$
- *Associativity:* $\forall\, a, b, c \in G \,.\, (a \circ b) \circ c = a \circ (b \circ c)$
- *Existence of a neutral element n:* $\exists\, n \in G\ \forall\, a \in G \,.\, n \circ a = a \circ n = a$
- *Existence of inverse elements:* $\forall\, a \in G\ \exists\, a^{-1} \in G \,.\, a \circ a^{-1} = a^{-1} \circ a = n$

Definition 2 (Commutative group). *A group $(G, \circ)$ is a* commutative group *(also called* Abelian group*) if the following is satisfied:*

- *Commutativity:* $\forall\, a, b \in G \,.\, (a \circ b) = (b \circ a)$

Definition 3 (Order of a group). *The* order of a group $(G, \circ)$ *is the cardinality of its set, denoted as G. It represents the number of elements in G. If $|G| < \infty$, the group is called a finite group.*

In the following, the quotient set is defined, which is necessary for the group definition of the 3×3×1-Cube. This is based on the definition of equivalence relations and classes. Equivalence relations can be used to describe the relationships between two elements of a set and equivalence classes are sets that contain equivalent elements.

[2] `https://jperm.net/3x3/moves`, Rubik's Cube Move Notation, accessed 15.09.2023

Definition 4 (Equivalence relation). *A relation* $\sim \subseteq A \times A$ *is called an* equivalence relation *on* A *if the following three properties hold:*

- *Reflexivity:* $\forall\, x \in A\,.\, x \sim x$
- *Symmetry:* $\forall\, x, y \in A\,.\, (x \sim y) \Rightarrow (y \sim x)$
- *transitivity:* $\forall\, x, y, z \in A\,.\, (x \sim y) \wedge (y \sim z) \Rightarrow (x \sim z)$

Definition 5 (Equivalence classes). *Let* $\sim$ *be an equivalence relation on the set* M *and* $x \in M$*. The* equivalence class *of* x *is then written as* $[x]$ *and is defined as* $\{y \in M \mid y \sim x\} \subseteq M$*.*

Each element of the equivalence class $[x]$ is called a representative of $[x]$. All representatives of $[x]$ are equivalent to x.

Definition 6 (Quotient set). *The* quotient set $M/\sim$ *of the set* M *with respect to the equivalence relation* $\sim$ *consists of all equivalence classes and is defined as* $\{[x] \mid x \in M\}$*.*

Definition 7 (Permutation). *Let* M *be a set with* n *elements. A* (n-digit) permutation *is a bijective mapping* $\pi : M \to M$*.*

A permutation is the changing of the order of objects. For example, let M be a set with five objects: $1, 2, 3, 4$ and 5. Now these objects can be arranged in five slots. The slots are numbered 1 to 5. Then a function $\pi : \{1, 2, 3, 4, 5\} \to \{1, 2, 3, 4, 5\}$ can be defined where $\pi(i)$ is the object that is in slot i. If the objects are arranged in the order 5 1 4 3 2, 5 is in slot 1, 1 is in slot 2, etc. This is also shown in Figure 8 with the function π for this permutation (π can also be written as an assignment of the form $i \mapsto j$ for $\pi(i) = j$). This results in the following cyclic

Slot	1	2	3	4	5
Object	5	1	4	3	2

$$\pi(1) = 5, \pi(2) = 1, \pi(3) = 4, \pi(4) = 3, \pi(5) = 2$$

$$1 \mapsto 5, \quad 2 \mapsto 1, \quad 3 \mapsto 4, \quad 4 \mapsto 3, \quad 5 \mapsto 2$$

Fig. 8. Visualization of the permutation function π.

notation: $\pi = (1\ 5\ 2)\ (3\ 4)$. The cycle notation is read as follows: 1 maps to 5, 5

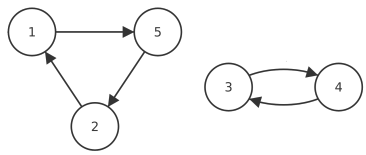


Fig. 9. A graphic presentation of the cycles of $\pi = (1\ 5\ 2)\ (3\ 4)$ is shown.

maps to 2, and 2 maps back to 1. This closes the first cycle. In the second cycle, 3 maps to 4, and 4 maps to 3. π consists of two cycles: one cycle has length three, and the other has length two. The cycles are graphically represented in Figure 9. The identity permutation is written as $\pi = 1$. In this case, all elements are mapped to themselves, and the cycles remain unchanged.

For n distinct objects, the number of possible permutations is $n!$.

Definition 8 (Composition of permutations). *Let π and τ be two n-digit permutations with $n \in \mathbb{N}$. The* successive execution of these permutations *is written as $\tau \diamond \pi$. Here π is executed first and τ is applied to the result.*

Let π and τ be permutations with $\pi = (1\ 2)\ (3\ 5)\ (4)$ and $\tau = (1\ 4\ 3)\ (2\ 5)$. The composition of the cycles $\tau \diamond \pi$ results in the following:

$$\tau \diamond \pi = (1\ 4\ 3)\ (2\ 5) \diamond (1\ 2)\ (3\ 5)\ (4) = (1\ 5)\ (2\ 4\ 3)$$

The cycle (1 5) is obtained as follows: In permutation π, 1 maps to 2, and in τ, 2 then maps to 5. This results in the cycle start (1 5 ...). In permutation π, 5 maps to 3, and 3 in τ maps to 1, which leads to the cycle (1 5). For the cycle (2 4 3), consider 2 in π. Since it is in the cycle (1 2), 2 maps to 1. In τ, 1 in the cycle (1 4 3) maps to 4 and (2 4 ...) is obtained. The 4 maps to itself in permutation π as it is a single-element cycle. In τ, 4 maps to 3, resulting in the cycle (2 4 3 ...). The 3 maps to 5 in permutation π, and 5 in τ maps to 2 and the cycle (2 4 3) is obtained. The composition of π and τ is illustrated in Figure 10 with arrows.

$$\pi \begin{cases} 1 & 2 & 3 & 4 & 5 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 2 & 1 & 5 & 4 & 3 \end{cases}$$

$$\tau \begin{cases} 2 & 1 & 5 & 4 & 3 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 5 & 4 & 2 & 3 & 1 \end{cases}$$

Fig. 10. The composition of π and τ is visually represented.

The following notations describe the same permutation: (1 3 4) (2) = (1 3 4) = (3 4 1) = (4 1 3).

4 Configurations of the Cube

To work with the cube, it needs to be determined in which configuration it is. The configuration of the 3×3×1-Cube is based on the configuration of the 3×3×3-Cube in [1]. For the 3×3×1-Cube, two components are used to determine the configuration:

- Position of the corner cubies (defined as function σ)
- Orientation of the edge cubies (defined as vector y)

The configuration C of the 3×3×1-Cube can be written as a 2-tuple: $C = (\sigma, y)$. No further components are needed, because the edge cubies do not change their position but only their orientation in the cube. They are attached in one position and can only be turned around, considering the cube mechanics. The orientation of the corner cubies is determined by their position and the corner cubies can have a visible side that does not have a colour panel (Figure 6).

Position of the corner cubies

The 3×3×1-Cube has four corner cubies. The set of the bijective permutation functions σ (for each basic rotation) contains functions, which represent the transitions of the corner cubies. The transitions describe the change in position of the cubies during a move. The names of the cube positions are shown in Figure 11. The individual positions are named with abbreviations, which describe their position with *right*, *left*, *front*, *back*, *up* and *down*.

While the edge cubies do not change their position, the corner cubies allow the shape shifting of the cube. This means, that it is also possible for the corner cubies to reach positions in a layer on top or below the cube. The names of these eight reachable positions in the top and bottom layer can be seen in Figure 11. The top and bottom layers are created due to the shape shifting of the cube. In the following, the permutation σ_R is derived, for a rotation of the right side

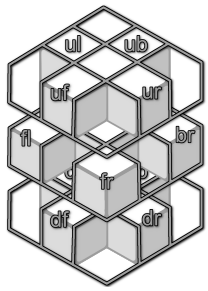


Fig. 11. All possible positions of the corner cubies are represented and named.

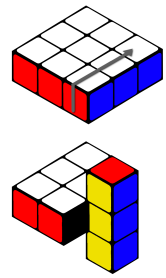


Fig. 12. Visual representation of the execution of move R.

by 90° in a clockwise direction. The derivation of the other rotations works analogously. The rotation of the right side is shown graphically in Figure 12. It can be seen that only the cubies in the places *ur*, *br*, *dr* and *fr* are permuted.

$$\sigma_R(fr) = ur \qquad \sigma_R(ur) = br \qquad \sigma_R(br) = dr \qquad \sigma_R(dr) = fr$$

The other cubies and the according permutation functions map to themselves, because this part of the cube does not get influenced by the move R.

$$\sigma_R(uf) = uf \qquad \sigma_R(ub) = ub \qquad \sigma_R(df) = df \qquad \sigma_R(db) = db$$
$$\sigma_R(fl) = fl \qquad \sigma_R(ul) = ul \qquad \sigma_R(dl) = dl \qquad \sigma_R(bl) = bl$$

This results in the following function σ_R:

$$\sigma_R = (fr\ ur\ br\ dr)\ (uf)\ (ub)\ (df)\ (db)\ (fl)\ (ul)\ (dl)\ (bl)\ = (fr\ ur\ br\ dr)$$

The position of the corner cubies in the start configuration is defined with the identity permutation $\sigma = 1$. The indices at the functions σ represent the different basic moves of the cube, which were defined in Figure 7. The identity permutation is written as $\sigma = 1$. This corresponds to the permutation of the

empty move. The permutation of the empty move is defined as $\sigma_N = 1$. The rotations of all sides can be described by the following cycles:

$$\begin{aligned} \sigma_F &= (uf\ fr\ df\ fl) & \sigma_B &= (ub\ bl\ db\ br) \\ \sigma_L &= (ul\ fl\ ul\ bl) & \sigma_R &= (fr\ ur\ br\ dr) \\ \sigma_N &= 1 \end{aligned}$$

Orientation of the edge cubies

The 3×3×1-Cube has four edge cubies with five colour panels each. In order to recognise the orientation of the cubies, the cube positions with a colour panel have a number assigned to them, which can be seen in Figure 13. These numbers are referred to as y_i with $i \in \{1, 2, 3, 4\}$ and each cubie has a number assigned on each colour panel. Because the *outer* colour panel cannot change its direction (only its orientations), it does not need to get a number assigned. The white colour panel starts with the number zero and then the numbers 2 and 3 get assigned clockwise (Figure 14).

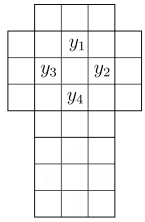


Fig. 13. The entries (y_1, y_2, y_3, y_4) of vector y of the edge cubies are marked.

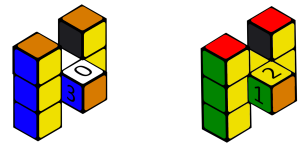


Fig. 14. The numbers on the colour panels of the edge cubies are shown.

The amount of numbers that are needed to identify the edge cubies orientation in each position is 4, because the four edge cubies have four possible positions in which they can be after executing a move on the 3×3×1-Cube. In the start configuration all y_i are 0, the vector y is $(0, 0, 0, 0)$, which can also be written as $y = 0$.

The transition function γ for the changes in vector y is defined for the execution of each basic move. The function f determines the new value of the new vector entries. This depends on the previous entry, which means it depends on the previous colour panel that was shown in the respective position. The derivation and definition of f is shown in the following. The function f is defined as $f(y) = (y + 1) \mod 4$.

$$\begin{aligned} \gamma_B((y_1, y_2, y_3, y_4)) &= (f(y_1), y_2, y_3, y_4) \\ \gamma_R((y_1, y_2, y_3, y_4)) &= (y_1, f(y_2), y_3, y_4) \\ \gamma_L((y_1, y_2, y_3, y_4)) &= (y_1, y_2, f(y_3), y_4) \\ \gamma_F((y_1, y_2, y_3, y_4)) &= (y_1, y_2, y_3, f(y_4)) \end{aligned}$$

The function f represents the transition for the entries in vector y for the edge cubies. In Figure 15, the four rotation options for an edge cubie can be seen.

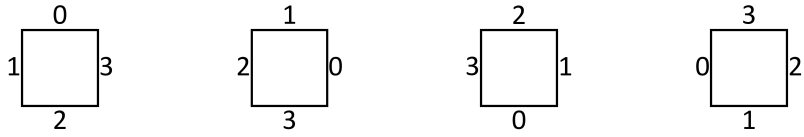


Fig. 15. Visualization of the rotation of an edge cubie with its assigned numbers.

Assuming, the respective entry in the vector y is on top, the entry would be 0 in the leftmost square, 1 in the second left square, then 2 and then 3 in the rightmost square, followed by 1 again after another rotation.

Because the sides of the cube are turned in clockwise direction, the following function occurs for the change of the position of the cubie:

$$f(y) = \begin{cases} 1 & \text{if } y = 0 \\ 2 & \text{if } y = 1 \\ 3 & \text{if } y = 2 \\ 0 & \text{if } y = 3 \end{cases} \qquad \begin{aligned} &f : \{0,1,2,3\} \to \{0,1,2,3\} \\ &f(y) = (y+1) \mod 4 \end{aligned}$$

Execution of moves

A cube configuration $C = (\sigma, y)$ can be changed by executing a move on the cube. The permutation σ then represents the new position of the corner cubies as a permutation starting from the positions in the start configuration and the vector y represents the orientation of the edge cubies. A move M can be one of the basic moves (*F, B, L, R*) or any sequence of basic moves. If a move M is executed on a cube configuration C, it can be written as $C \cdot M$. The result of $C \cdot M$ is then a new configuration C'.

As an example, the move R is executed on the start configuration $C = (1, 0)$. The execution of the move R was already shown in Figure 12. The vector y starts with $y = (0, 0, 0, 0) = 0$ in the start configuration and the permutation σ is the identity permutation.

$$\begin{aligned} & C \cdot M \\ \Rightarrow\ & (1,0) \cdot R \\ =\ & (1, (0,0,0,0)) \cdot R \\ =\ & ((\sigma_R \diamond 1), \gamma_R((0,0,0,0))) \\ =\ & ((\sigma_R \diamond 1), (0, f(0), 0, 0)) \\ =\ & ((\mathit{fr\ ur\ br\ dr}), (0,1,0,0)) = C' \end{aligned}$$

Equality of moves

The set A_M is defined as the infinite set of all moves of the 3×3×1-Cube. Two moves $M_1, M_2 \in A_M$ are considered equal if they produce the same configuration with the same initial configuration of the cube. A move consists of one or more basic moves (R, L, F, B). The multiple execution of moves can be represented with the exponent notation. For example, RR (two clockwise rotations of the right plane) is also written as R^2. M^0 represents an empty move, that causes no change in the configuration of the cube.

One example for equivalent moves is the repetition of one basic move four times – the cube ends in the same position as it started.

For any cube configuration C and all $n \in \mathbb{N}$ then holds:

$$\forall\ M \in \{R, L, F, B\}, n \in \mathbb{N}\ .\ C \cdot M^n = C \cdot M^{n \bmod 4} \text{ and}$$
$$\forall\ M \in \{R, L, F, B\}, n \in \mathbb{N}, n \bmod 4 = 0\ .\ C \cdot M^n = C$$

Since the number of possible moves ($|A_M|$) is infinite while the number of valid cube configurations is finite, there is an infinite amount of cases in which two moves cause the same following configuration of the cube and are considered equal.

To ensure that equal moves that end in the same configuration are represented as one element, an equivalence relation is introduced. In the case of the 3×3×1-Cube, it is a relation $\sim$ between two moves M_1, M_2 from the set of all moves A_M. The relation $\sim$ checks any two moves M_1 and M_2 for equality and is a subset of $A_M \times A_M$. The relation is defined below for all moves $M_1, M_2 \in A_M$:

$$M_1 \sim M_2 :=\ C \cdot M_1 = C \cdot M_2$$

For a relation to be an equivalence relation, the following three properties must hold: Reflexivity, symmetry, transitivity.

Reflexivity For reflexivity, $M \sim M$ must hold for all moves. The following expressions apply to all M from A_M and any cube configuration C.

$$M \sim M := C \cdot M = C \cdot M$$

Symmetry For symmetry, the following must hold for all M_1 and M_2 from A_M: If $M_1 \sim M_2$, then $M_2 \sim M_1$. In the following, C is an arbitrary cube configuration.

$$M_1 \sim M_2 \Rightarrow M_2 \sim M_1$$
$$\Leftrightarrow C \cdot M_1 = C \cdot M_2 \Rightarrow C \cdot M_2 = C \cdot M_1$$

Transitivity For transitivity, it must hold for all moves M_1, M_2, M_3 from A_M and a cube configuration C: If $M_1 \sim M_2$ and $M_2 \sim M_3$, then $M_1 \sim M_3$.

$$\begin{aligned} &M_1 \sim M_2 && \wedge M_2 \sim M_3 && \Rightarrow M_1 \sim M_3 \\ \Leftrightarrow\ &(C \cdot M_1 = C \cdot M_2) && \wedge (C \cdot M_2 = C \cdot M_3) && \Rightarrow (C \cdot M_1 = C \cdot M_3) \end{aligned}$$

Since the moves M_1 and M_2 result in the same cube configuration and the moves M_2 and M_3 also result in the same cube configuration, the relation $\sim$ holds for the two moves M_1 and M_3 as well.

The relation $\sim$ is considered an equivalence relation, assuming satisfaction of the properties of reflexivity, symmetry, and transitivity. A comprehensive proof lies beyond the scope of this paper.

The equivalence classes of the equivalence relation $\sim$ are defined on the set of all moves A_M as

$$[M] := \{M' \in A_M \mid M' \sim M\} \subseteq A_M$$

Each equivalence class $[M]$ contains all moves of the set A_M that are equivalent to the move M with the equivalence relation $\sim$. Hence, $[M]$ is a subset of A_M. All equivalent elements of an equivalence class $[M]$ are called representatives of this equivalence class. For example, the moves R and $RRRRR$ are representatives of the equivalence class $[R]$, since the moves R and $RRRRR$ are equivalent to R with the equivalence relation $\sim$.

Two equal moves $M_1, M_2 \in A_M$ are always representatives of the same equivalence class. Conversely, two moves $M_1, M_2 \in A_M$ that do not lead to the same following configuration are representatives of different equivalence classes. It follows that two different equivalence classes do not contain equivalent moves, since all equivalent moves are in exactly one equivalence class.

This leads to the quotient set $G_{3\times3\times1}$, which consists of equivalence classes that represent every move. It contains the equivalence classes of all moves without duplicating equal moves. The elements of $G_{3\times3\times1}$ are all equivalence classes with respect to the equivalence relation $\sim$.

$$G_{3\times3\times1} := \{[M] \mid M \in A_M\}$$

In this paper, the square brackets of the equivalence classes in $G_{3\times3\times1}$ are omitted to simplify the notation.

Two equivalence classes or their representatives can be linked by the $\circ$ operator as follows: Let M_1 and M_2 be two moves in $G_{3\times3\times1}$. Then $M_1 \circ M_2$ means that M_1 is executed first and then M_2. $M_1 \circ M_2$ can also be written as M_1M_2.

5 Cube as a group

The group of the 3×3×1-Cube is called $(G_{3\times3\times1}, \circ)$. The set $G_{3\times3\times1}$ of the group contains the equivalence classes of all possible moves of the cube. The operator $\circ$ is defined as the concatenation of two moves. In the following, the group properties are examined to show that $(G_{3\times3\times1}, \circ)$ is a group.

Closure

$$\forall\ M_1, M_2 \in G_{3\times3\times1}\ .\ (M_1 \circ M_2) \in G_{3\times3\times1}$$

Proof (Closure). Let M_1 and M_2 be moves and thus elements of $G_{3\times3\times1}$. Then $M_1 \circ M_2$ is also an element of $G_{3\times3\times1}$. The set $G_{3\times3\times1}$ contains the equivalence classes of the elements from the infinite set of all moves A_M. Because of this, all elements of the set A_M are representatives of the equivalence classes in $G_{3\times3\times1}$. Consequently, the group $(G_{3\times3\times1}, \circ)$ is closed under the operator $\circ$. □

Associativity

$$\forall\ M_1, M_2, M_3 \in G_{3\times3\times1}\ .\ (M_1 \circ M_2) \circ M_3 = M_1 \circ (M_2 \circ M_3)$$

Proof (Associativity). For this proof, a notation for executing moves is introduced: An arbitrary but fixed cubie is here called s. When executing a move M, $M(s)$ can be written to obtain the new position of the cubie.

Considering $M_1 \circ M_2$, M_1 is executed first and then M_2. M_1 moves the cubie s to the position $M_1(s)$ and then M_2 moves the cubie to the position $M_2(M_1(s))$. Consequently, $M_1 \circ M_2 = M_2(M_1(s))$. Now $(M_1 \circ M_2) \circ M_3 = M_1 \circ (M_2 \circ M_3)$ has to be shown. To do this, it is shown that $(M_1 \circ M_2) \circ M_3$ and $M_1 \circ (M_2 \circ M_3)$ can both be transformed into $M_3(M_2(M_1(s)))$:

$$\begin{aligned} & (M_1 \circ M_2) \circ M_3 \\ \Leftrightarrow\ & ((M_1 \circ M_2) \circ M_3)(s) \\ =\ & M_3((M_1 \circ M_2)(s)) \\ =\ & M_3(M_2(M_1(s))) \end{aligned} \qquad \begin{aligned} & M_1 \circ (M_2 \circ M_3) \\ \Leftrightarrow\ & (M_1 \circ (M_2 \circ M_3))(s) \\ =\ & (M_2 \circ M_3)(M_1(s)) \\ =\ & M_3(M_2(M_1(s))) \end{aligned}$$

Consequently, the group $(G_{3\times3\times1}, \circ)$ is associative. □

Existence of a neutral element

$$\exists\ N \in G_{3\times3\times1}\ \forall\ M \in G_{3\times3\times1}\ .\ N \circ M = M \circ N = M$$

Proof (Existence of a neutral element). For the neutral element $N \in G_{3\times3\times1}$ and all moves $M \in G_{3\times3\times1}$ must hold: $N \circ M = M \circ N = M$. If the move M is connected with the neutral element N, it remains unchanged.

Considering the physical 3×3×1-Cube, the neutral element of the group $(G_{3\times3\times1}, \circ)$ is the empty move. None of the sides of the cube are being rotated. If a move M is executed followed by the move N, this means *execute M first and then nothing*, which is the same as *execute M*.

If this is applied to the mathematical model of the cube, executing N does not change the configuration of the cube, which leads to $\sigma_N = 1$ and $\gamma_N(y) = y$.

For the neutral element N, $C \cdot N = C$ must hold for each cube configuration C. The configuration C remains in its initial configuration. This applies for example to rotating a single side four times. These four executions of a basic move are included in the equivalence class of N as representatives of the empty move. For any move M from $G_{3\times3\times1}$, $M^0 = N$ is true, since the cube configuration remains unchanged when a move is executed zero times.

The neutral element N of $(G_{3\times3\times1}, \circ)$ is the equivalence class of the empty move, that contains every equivalent move to the empty move. □

Existence of inverse elements

$$\forall\ M \in G_{3\times3\times1}\ \exists\ M^{-1} \in G_{3\times3\times1}\ .\ M \circ M^{-1} = M^{-1} \circ M = N$$

Proof (Existence of inverse elements). Each move M transforms a cube configuration into a new configuration. Looking at the physical 3×3×1-Cube, one may notice that the individual sides can be rotated counter-clockwise to invert a single rotation. A rotation of 90° counter-clockwise corresponds to a plane rotation of 270° clockwise, which is equivalent to three clockwise turns. The inverses of the individual rotations are defined as follows:

$$\forall\ M \in \{R, L, F, B\}\ .\ M^{-1} = MMM = M^3$$

If a move consisting of several basic moves has to be inverted, the last element must be inverted first. For a move P of length n, the n-th basic move M_n is inverted first by transforming it into $M^{-1} = M_n M_n M_n$. Then the other elements of the complete move are inverted from the rear. In the following, the inverse element for each move is defined, with $n \in \mathbb{N}$ and M_i representing all basic moves M_1 to M_n that make up the move P.

$$\forall\ P \in G_{3\times3\times1}, P = (M_1 \circ M_2 \circ ... \circ M_n), M_i \in \{R, L, F, B\}\ .$$
$$P^{-1} = {M_n}^{-1} \circ {M_{n-1}}^{-1} \circ ... \circ {M_1}^{-1}$$

With these expressions, the inverse element for each move M from the set $G_{3\times3\times1}$ is defined. □

$(G_{3\times3\times1}, \circ)$ is a group since the four group axioms are satisfied.

Checking for commutativity

$(G_{3\times3\times1}, \circ)$ is not a commutative group. This can be shown with a counter example: A clockwise rotation of the right side (R) and a clockwise rotation of the front side (F) in reverse order on the solved cube have different results, which means $C \cdot FR \neq C \cdot RF$ for any cube configuration C. This example is shown graphically with C as the start configuration $(C = (1, 0))$ in Figure 16.

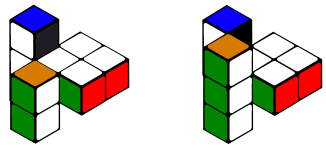


Fig. 16. The execution of the move FR on the left and RF on the right can be seen.

The cube configurations are shown in the following, where $C_{RF} \neq C_{RF}$:

$$
\begin{aligned}
& C_{FR} = C \cdot FR \\
\Rightarrow\ & (1,0) \cdot FR \\
=\ & ((\sigma_F \diamond 1), (0,0,0,f(0))) \cdot R \\
=\ & ((ul\ fr\ df\ fl), (0,0,0,1)) \cdot R \\
=\ & (\sigma_R \diamond (ul\ fr\ df\ fl), (0, f(0), 0, 1)) \\
=\ & ((fr\ ur\ br\ dr) \diamond (uf\ fr\ df\ fl), (0,1,0,1)) \\
=\ & ((fr\ df\ fl\ uf\ ur\ br\ dr), (0,1,0,1))
\end{aligned}
\qquad
\begin{aligned}
& C_{RF} = C \cdot RF \\
\Rightarrow\ & (1,0) \cdot RF \\
=\ & ((\sigma_R \diamond 1), (0, f(0), 0, 0)) \cdot F \\
=\ & ((fr\ ur\ br\ dr), (0,1,0,0)) \cdot F \\
=\ & (\sigma_F \diamond (fr\ ur\ br\ dr), (0,1,0,f(0))) \\
=\ & ((uf\ fr\ df\ fl) \diamond (fr\ ur\ br\ dr), (0,1,0,f(0))) \\
=\ & ((fr\ ur\ br\ dr\ df\ fl\ uf), (0,1,0,1))
\end{aligned}
$$

6 Order

Order of the group

The order of the group $(G_{3\times3\times1}, \circ)$ is the number of elements of $G_{3\times3\times1}$ (also written as $|G_{3\times3\times1}|$). The order of the group is equal to the number of valid cube configurations, since $G_{3\times3\times1}$ contains the equivalence classes of all possible moves. Thus, each element of $G_{3\times3\times1}$ changes the cube into a different cube configuration and no equivalent moves are included twice. Furthermore, the equivalence classes were formed with the set of all moves of the cube. Thus, every possible move is included and $|G_{3\times3\times1}|$ corresponds to the number of valid cube configurations.

Since the order of the group $(G_{3\times3\times1}, \circ)$ is equal to the number of valid cube configurations, it is composed of the product of the two configuration parameters σ and y.

σ is the set of bijective functions for each rotation. It contains the transitions of the four corner cubies as permutations. Since there are four permutation objects and 12 possible positions, the number of permutations is calculated with the expression $\frac{12!}{(12-4)!}$.

The other parameter of the cube configuration is the 4-dimensional vector y. Each of the values in y is from the set $\{0,1,2,3\}$. Thus, there are four possibilities for each value in the vector, which leads to 4^4 possible vector configurations. This can be combined as follows:

$$\frac{12!}{(12-4)!} \cdot 4^4 = \frac{12!}{8!} \cdot 256 = \frac{479\ 001\ 600}{40\ 320} \cdot 256 = 11\ 880 \cdot 256 = 3\ 041\ 280$$

The 3×3×1-Cube has 3 041 280 possible configurations.

Optimal solution

For finding the optimal solution for the 3×3×1-Cube in any configuration, two trees are modelled and compared. The maximum depth of these trees is the God's Number of the cube [9]. The God's Number indicates the maximum number of moves needed to solve the cube when using the optimal solution path. The God's

Number of the 3×3×1-Cube has not been determined yet and doing so would exceed the scope of this paper.

A tree for finding the optimal solution contains an arbitrary configuration C as root. Starting from this node, there is an edge for each possible move of the cube. These edges each lead to a subsequent configuration C_i, where the index i contains the previous moves of the path in the tree. Each node C_i is followed by a node C_{iM} in the next level via an edge of a move M. The path i, which represents a solution path, leads to a node with the configuration $C_i = (1, 0)$. To determine the length of a path, the number of level rotations denoted in the index i must be determined as $|i|$. A configuration C_i is reached after $|i|$ moves. This allows the length of the optimal solution path to be read out.

The group $(G_{3\times3\times1}, \circ)$ is based on the four moves R, L, F and B. But each side on the physical cube can be rotated in two directions, which leads to eight possible moves: R, R^{-1}, L, L^{-1}, F, F^{-1}, B and B^{-1}. In a tree T_8, each of the tree nodes therefore has eight children (Figure 17). Since the God's Number of the 3×3×1-Cube is not determined yet, the variable x_{GN} is used as a placeholder: The cube is solved after a maximum of x_{GN} moves. The cube configuration is then $C_i = (1, 0)$ with $|i| \leq x_{GN}$ and $|i|$ corresponds to the depth of the tree T_8. Accordingly, the tree has a maximum size of $8^{x_{GN}}$ nodes.

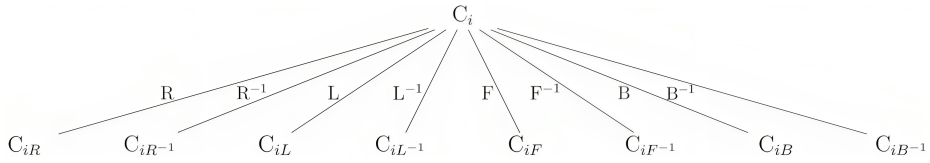


Fig. 17. Tree T_8 for finding the optimal solution of the 3×3×1-Cube, where each node has eight children.

To design a tree T_4 with a minimum number of children, the edges R^{-1}, L^{-1}, F^{-1} and B^{-1} are omitted (Figure 18). These moves can be represented by RRR, LLL, FFF and BBB and each node has only four children then. The moves R^{-1}, L^{-1}, F^{-1} and B^{-1} are each represented by three edges. This results in a maximum depth of $|i| = 3 \cdot x_{GN}$. The maximum size of T_4 is therefore $4^{3x_{GN}}$.

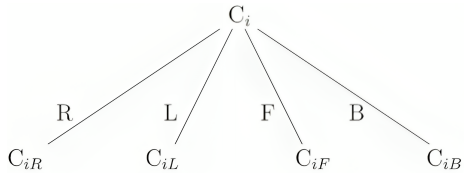


Fig. 18. Tree T_4 for finding the optimal solution of the 3×3×1-Cube, where each node has four children.

For every $x_{GN} \in \mathbb{N}$ as a possible God's Number, the amount of nodes in the tree T_4 is significantly larger than in tree T_8 ($8^{x_{GN}} \leq 4^{3x_{GN}}$ for all $x_{GN} \in \mathbb{N}$). Because of this, tree T_8 with eight children per node is more suitable for

calculating the optimal solution path, as it can be calculated more efficiently due to its smaller number of nodes. In addition to this, T_4 has a larger or equal number of nodes along each path than T_8.

7 Conclusion

In this paper, the group theory of the 3×3×3-Cube was transferred to the 3×3×1-Cube. The 3×3×1-Cube sets apart from the 3×3×3-Cube due to its shape shifting possibilities. The mechanics of the cubes differ, which leads to different requirements for the mathematical representation of the different cubes.

The group $(G_{3\times3\times1}, \circ)$ of the 3×3×1-Cube consists of the set of moves $G_{3\times3\times1}$ and the operator $\circ$ that combines two moves. The set $G_{3\times3\times1}$ is a quotient set and contains the equivalence classes of all moves, representing equivalent moves by a single element. Since each cube configuration can be reached by a different move, and the set $G_{3\times3\times1}$ contains all equivalent moves in a single equivalence class, the cardinality of $G_{3\times3\times1}$ is equal to the number of valid cube configurations. There are 3 041 280 valid configurations for the 3×3×1-Cube.

In addition to this, approaches to solving the cube were described and the role of the God's Number in this was discussed. Tree concepts for finding an optimal solution path were introduced.

The group of the 3×3×1-Cube illustrates group theory through a complex yet tangible example. Various concepts were applied, including group theory, equivalence relations and classes, quotient sets, group cardinality, permutations and cycle notation.

Future work

In this work, the group theory of the 3×3×1-Cube was described and proven and based on this, future opportunities emerge that are discussed in the following.

The group's generators and their Cayley graphs are another interesting aspect to examine. Generators are elements of a group that can generate the entire group through combining themselves with the group operator and Cayley graphs are graphical representations used to visualise the structure of a group by assigning nodes to the group elements and connecting them with edges corresponding to group operations. Here, the nodes would represent each possible cube configuration while the edges would embody the moves that can be executed on the cube, leading to a following cube configuration. Through the Cayley graph, the cyclic behaviour of moves can be visualised. This cyclic behaviour can also be examined by analysing the order of the permutations and the moves. Cayley graphs can also contribute to a better understanding of the structure of a group.

The definition of the equality of two moves and the equivalence classes of moves lead to the challenge of calculating the equality of any two specific moves. This can be implemented using a term rewriting system and is necessary for generating the complete Cayley graph of the group.

Additionally, commutators can obtain meaningful results regarding the commutativity of pairs of moves. Commutators can measure the deviation from commutativity, revealing the non-commutative aspects between pairs of elements within a group. For this, a larger set of commutators needs to be calculated and compared. The computation and evaluation of commutators can be automated.

Based on the group-theoretical model of the 3×3×1-Cube, a *GUI* (Graphical User Interface) for solving the cube can be implemented with a backend, that stores and processes all the information about the cube. The optimal solution for the cube can be calculated using the tree concept that was described in this paper. This can lead to finding and evaluating efficient algorithms for solving the cube and various solving methods can be compared and optimised.

To gain more information, not only about the depth of the trees for the minimal solution but about the 3×3×1-Cube in general, the calculation of the God's Number presents an exciting challenge.

References

1. Janet Chen. Group Theory and the Rubik's Cube, 2004. Mathematics course notes from Harvard University.
2. Claire Cornock. Teaching group theory using Rubik's cubes. *International Journal of Mathematical Education in Science and Technology*, 46(7):957–967, 2015.
3. Erik Demaine, Martin Demaine, Sarah Eisenstat, Anna Lubiw, and Andrew Winslow. Algorithms for solving Rubik's cubes. In *Algorithms–ESA 2011: 19th Annual European Symposium, Saarbrücken, Germany, September 5-9, 2011. Proceedings 19*, pages 689–700. Springer, 2011.
4. Daniel Duberg and Jakob Tideström. Comparison of Rubik's Cube Solving Methods Made for Humans, 2015. Degree project in Computer Science at CSC School Stockholm.
5. Tobias Glosauer. *Elementar(st)e Gruppentheorie*. Springer Spektrum, Wiesbaden, 2016.
6. David Joyner. *Adventures in Group Theory: Rubik's Cube, Merlin's Machine, and Other Mathematical Toys*. Johns Hopkins University Press, Baltimore, USA, 2008.
7. Pina Kolling. Gruppentheorie des 2×2×2 Zauberwürfels und dessen Lösungsalgorithmen, 2021. Bachelor's thesis in Computer Science at TU Dortmund University.
8. OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving Rubik's Cube with a Robot Hand. *arXiv preprint*, 10 2019.
9. Tomas Rokicki. Towards God's Number for Rubik's Cube in the Quarter-Turn Metric. *The College Mathematics Journal*, 45(4):242–253, 2014.
10. Tomas Rokicki, Herbert Kociemba, Morley Davidson, and John Dethridge. The Diameter of the Rubik's Cube Group is Twenty. *siam REVIEW*, 56(4):645–670, 2014.

The Effect of Gender on ChatGPT Use: A Technology Acceptance Model 3 (TAM3) study

Susan Kronberg

Department of Computing Science
Umeå University, Sweden
id19skg@cs.umu.se

Abstract. This study investigates gender disparities in ChatGPT use. Framed by the research question, "Does gender impact the use of ChatGPT?" a survey including an experiment and questions based on the Technology Acceptance Model 3 was conducted among students in Sweden. Notably, only 7 out of 61 data points showed a difference. Differences were found in current ChatGPT use, with male participants reporting higher frequency, while no significant disparities were observed in the experiment or future intent to use. The study reveals small differences in AI anxiety within a homogeneous group characterized by high technology experience and a dissonance between perceptions of quality and and accuracy of ChatGPT results. These insights contribute to the literature on the effect of gender on technology acceptance and highlight the need for a deeper understanding of the nuanced dynamics surrounding the use of and perceptions of generative AI technologies like ChatGPT.

1 Introduction

There is an ongoing surge in generative artificial intelligence (AI) applications. From DALL-E[1] and Midjourney[2] for text-to-image generation to Harvey AI[3] for law professionals and Jasper AI[4] for marketing and content creation, among others. One standout in this rapid expansion is ChatGPT 3.5 which has become one of the fastest-growing applications to date, gaining over 100 million users within the first 2 months of release [13].

ChatGPT 3.5 is a natural language processing (NLP) model that can understand and generate text. Its vast knowledge base and ability to generate compelling human language make it a powerful tool to streamline tasks and processes [9]. There are however concerns in the literature regarding GPT, and generative artificial intelligence (AI) in general, such as the perpetuation of gender, racial, and political biases [8], as well as privacy and integrity concerns [14].

[1] https://openai.com/dall-e-2 DALL-E 2 homepage, accessed 2024-01-05
[2] https://www.midjourney.com Midjourney homepage, accessed 2024-01-05
[3] https://www.harvey.ai/ Harvey AI homepage, accessed 2024-01-05
[4] https://www.jasper.ai/ Jasper AI homepage, accessed 2024-1-05

Suna Bensch (Eds.): Umeå's 27th Student Conference in Computing Science USCCS 2024, pp. 57–72, January 2024.

While the user base for ChatGPT is steadily growing, the literature shows gender disparities when it comes to the acceptance of new technology; the acceptance of cryptocurrency being such an example [1]. Studies have shown that women tend to display higher levels of technology aversion than men [7]. This poses the question if this is a trend that holds in the acceptance of generative AI, such as ChatGPT. Globally, there is currently a lack of women in research and development of AI, as well as a gender gap in general digital skills, where women are "25% less likely than men to know how to leverage digital technology for basic uses" [17] (p.3), with this gap in digital skills growing in high-income countries. Furthermore, many female-dominated jobs such as administration and customer support are predicted to be automatized [17]. Considering the potential of generative AI technology to increase users' productivity and enhance learning for novices in different domains [3], it may lead to a situation where non-AI adopters are left behind, thus it is imperative to understand the mechanisms behind AI acceptance and adaptation.

This paper aims to explore potential gender disparities in the use of ChatGPT posing the following research question *Does gender impact the use of ChatGPT*? A survey was conducted using the Technology Acceptance Model 3 (TAM3) [18] to assess the acceptance of ChatGPT amongst students in Sweden. The survey included an experiment in which participants were introduced to three different scenarios regarding different tasks/problems which could be solved using ChatGPT or a traditional method, such as generating a text manually or finding information through Google. The purpose of the experiment was to observe when users preferred using ChatGPT.

The results revealed some differences in the current use of ChatGPT between the male and female participants, where the male participants reported using ChatGPT more frequently. Notably, no significant difference was found in the experiment or questions about future intent to use. Although indicators of AI anxiety were generally low, the findings indicate that the male participants exhibited lower levels, despite the target group being homogeneous in terms of high technical experience. The results also indicate a dissonance in the perceptions of result quality and result accuracy, where the quality of ChatGPT results where perceived to be high but trust in the accuracy was generally low. These findings suggest that there may exist other factors affecting perceptions of AI systems, which could be potential avenues for future research.

2 Literature review

In this section, the complementary concepts of technology aversion and technology acceptance are explored. Understanding these concepts helps us gain insight into driving factors that influence the use and adaptation of new technology.

2.1 Technology aversion

Technology aversion, algorithmic aversion, technophobia, and technology anxiety all aim to examine negative perceptions and emotions towards technology [7, 5].

Such negative perceptions and emotions have been shown to appear amongst users of technology from computers and the internet [2], to algorithms and AI [11]. Algorithmic aversion has been described as the phenomenon in which people choose a human agent over an algorithmic agent, even if the algorithmic agent outperforms the human counterpart [6].

The acceptance of AI evaluators in the job market, as opposed to human evaluators, has been studied with regard to gender [15]. The results showed that people preferred human evaluators regardless of the gender of the participants and the gender of the human evaluators. It was however shown that women who perceived AI as able to reduce bias in the hiring process preferred AI at a higher rate. This is an example of algorithmic aversion at play. Another study showed that if a forecast algorithm makes mistakes people are less likely to trust the algorithm even if it is generally more accurate than human forecasters [5]. On the other hand, people will be more likely to use algorithms if they can modify them, even slightly [6]. Since ChatGPT users can modify the output of ChatGPT by prompting, this may have a positive effect on its acceptance.

Technology anxiety and technophobia involve a heightened level of discomfort regarding technology, manifesting in anxiety or fear. These heightened levels of negative emotions can be expressed as anxiety about using technology or fear of the technology itself, what it can achieve, and the potential consequences that technology may have on society [7].

Demographic variables play an intricate role in developing technology anxiety among users. Variables such as age, gender and academic background have been shown to correlate with technology anxiety. However, individual factors, specifically experience with technology and self-efficacy have, a moderating effect on technology anxiety among users [7]. The fear of autonomous robots and AI (FARAI) has been studied, and while demographic variables such as age, gender, and education all were small but significant predictors of FARAI, exposure to science fiction media was a moderate predictor [12].

2.2 Technology acceptance

In contrast to technology aversion, there is the concept of technology acceptance, which focuses on factors that may impact the acceptance and adaptation of technology. The original Technology Acceptance Model (TAM) is a model for predicting individuals' adoption of technology through *Behavioural Intent to Use* (BI). It introduces the determinant variables *Perceived Usefulness* (PU) and *Perceived Ease of Use* (PEOU) as fundamental for use intention and acceptance of new technology [4]. It has then been extended into the Technology Acceptance Model 2 (TAM2) which includes *social influence processes* and *cognitive instrumental processes* as factors that affect perceived usefulness [19]. Social influence processes aim to describe how others can influence the acceptance of a technology, through subjective norms, image, and voluntariness. The cognitive instrumental processes in TAM2 relate to the perceptions of how well a system matches one's specific job, tasks or goals, thus affecting the perceived usefulness

of the system. The cognitive instrumental processes in TAM2 are job relevance, output quality, and result demonstrability.

TAM3 is the third iteration of the model and has been extended to include computer anxiety, computer playfulness, computer self-efficiency, perception of external control, computer enjoyment, and objective usability, that is the actual observed level of ease of use, instead of the perception of it [18, 4]. All variables used in TAM3 are described in Table 1.

The role of social influence, perceived usefulness, facilitating condition, perceived ease of use, and AI aversion has been studied when it comes to adopting AI-based tools in social development organizations in India, specifically AI tools for collaboration between employees [10]. The findings support that the adoption of AI tools is influenced by perceived usefulness. Furthermore, the findings support past literature on the negative correlation between AI aversion and AI use, as there is a decrease in AI tool use as aversion towards AI increases. This study did however not examine the impact of gender.

3 Method

A quantitative approach was used to examine whether any gender differences exist in the acceptance of ChatGPT. The study consisted of an online survey which included a pre-questionnaire where demographic information was collected, an experiment to gain insight into when users prefer ChatGPT to solve problems compared to traditional problem-solving methods, and a post-questionnaire using TAM3 [18] to gather information about perceptions and current ChatGPT usage.

Twelve participants were recruited from the 5-year MSc program in Interaction Technology and Design at Umeå University through convenience sampling, see Table 2 for demographics distribution. The participants had a homogeneous academic background and similar technical experience. Considering that technical experience has been shown to be a moderating variable for technology aversion [7], a homogeneous sample concerning technical experience reduces such variability, making the gender-related variables easier to compare.

Before starting the survey, each participant was asked for consent to contribute to the study. Anonymity was ensured and the voluntariness of contributing was communicated. Furthermore, they were made aware of the study's purpose of investigating the impact of gender on ChatGPT usage for the sake of transparency, as we saw no negative impact of disclosing such information.

3.1 Experiment Design

The survey's experimental part examines three scenarios in which users may or may not use ChatGPT to solve different problems, and provides the opportunity to examine individuals' perceived usability through observation. Each scenario was designed such that the motive and task were clear at hand, see Figure 1. The scenarios relate to different types of tasks that can be accomplished using

Table 1: Descriptions of the variables in the extended Technology Acceptance Model (TAM3).

Variable	**Description**
Behavioral Intent of Use (BU)	Measures the intention to use a technology in the future.
Perceived Usefulness (PU)	"The degree to which a person believes that using a particular system would enhance his or her job performance." [4] (p.320)
Perceived Ease of Use (PEOU)	"The degree to which a person believes that using a particular system would be free of effort."[4] (p.320)
Subjective norm (SN)	Measures external influence by someone important to the user to use a technology [19].
Image (IMG)	How a person might use a technology if they believe that it will increase their social status or social image [19].
Voluntariness (VOL)	If the intent to use technology is perceived as mandatory or voluntary [19].
Job relevance (REL)	The perception of how relevant the technology is to a specific job or a task [19].
Output Quality (OUT)	The perception of the quality of the output of a system such as how accurate and reliable the outputs of are perceived to be [19].
Result Demonstrability (RES)	How the technology is perceived to have an impact on or benefit one's job performance [19].
Computer Anxiety (CANX)	An individual's level of anxiety or fear when thinking of or interacting with computers [18].
Computer Playfulness (CPLAY)	The perceived level of playfulness when interacting with computers [18].
Computer Self-efficiency (CSELF)	The level of one's perceived ability to use a computer to accomplish a task [18].
Perception of External Control (PEC)	The perception of having the resources, technical or organisational, to use the system [18].
Perceived Enjoyment (ENJ)	The level of enjoyment an individual may feel when using a system, related to the activity of using the system rather than its related outcomes [18]. .
Objective Usability (OU)	Observed usability rather than perceptions [18].
Use (USE)	Reported levels of use [18]

Table 2: Participant Demographics of Students Enrolled in 5-Year Programs at Umeå University

Category	**Number**	**Percentage (%)**
Gender	12	
Female	7	58.3
Male	5	41.7
Age Range	12	
18-24	8	66.7
25-30	4	33.3
Academic Background	12	
5th-year students	10	83.3
4th-year students	1	8.3
3rd-year students	1	8.3

ChatGPT [14]: search and recommendations in Figure 1a, learning in Figure 1b, and text generation in Figure 1c. The participants were requested to choose a method for solving the task, with a traditional method such as Google or manually writing, and using ChatGPT. An "other" option was also provided, allowing participants to provide an open-ended response. However, during the review and data cleaning process for statistical analysis, it was realized that some participants used the "other" option to indicate mixed use of the other options. These results were categorized as ChatGPT usage, as the focus was on the use or non-use of ChatGPT.

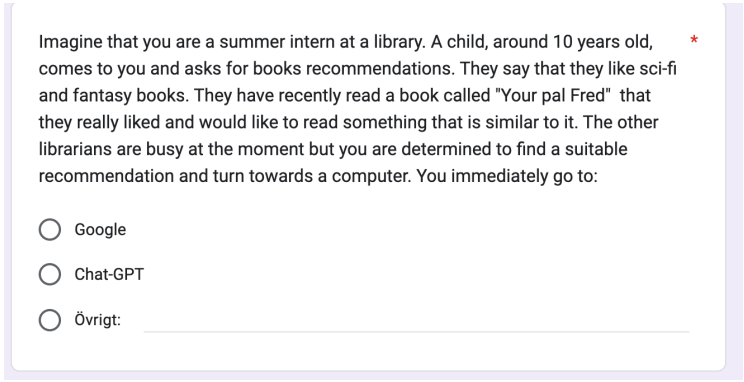
Each scenario was controlled for familiarity with the topic at hand through a 5-point scale ranging from "No experience" to "Advanced experience" with some variation on the formulation to better suit each topic. In total, the experimental part of the survey consisted of 8 questions.

3.2 Post-questionnaire

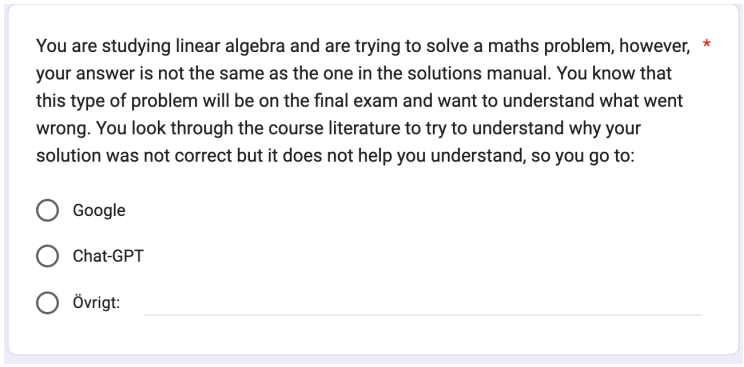
After completing all three scenarios, participants answered a Likert scale questionnaire based on TAM 3. Data regarding the participants' current use of ChatGPT was gathered in this section. They were also requested to try ChatGPT if they had not already done so before, as earlier experience with the system is critical to answering the TAM3 items. They were not asked to test ChatGPT before this stage to not impact their initial response to the experiment.

The questionnaire consisted of 51 items from the TAM3 model [18] using a 5-point Likert scale, with 3-4 items for each variable in Table 1 except for Objective Usability. Some items were rewritten to better reflect the student's academic context as opposed to working individuals in an organization. Other items were rephrased to suit the study of AI acceptance rather than traditional computer software.

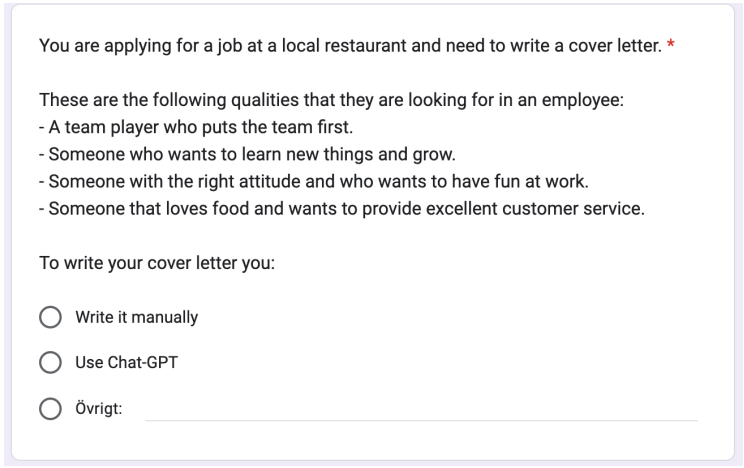
One item was added to the Output Quality items: "I trust that the results obtained by using ChatGPT are accurate", motivated by ChatGPT suffering from the hallucination effect and perpetuation of biases [8]. The hallucination effect is a phenomenon common in Large Language Models (LLM) where terms



(a) Description of a search and recommendation task. Related topics: science fiction and children's literature



(b) Description of a learning task. Related topics: maths and linear algebra



(c) Description of a writing task. Related topics: earlier experiences within the specific industry and cover letter writing experience

Fig. 1: Screenshots of the experimental part of the survey, where each scenario is described followed by the response options

and information are "invented" when generating a response [16], which affects the accuracy of the results.

3.3 Data analysis

The collected data was analyzed using ANOVA and the Tukey Test to analyze the significant variance differences obtained from ANOVA. 61 data points were used in the analysis. One from the pre-questionnaire for confirming homogeneity in regards to AI knowledge, 8 from the experimental section, and 52 from the post-questionnaire.

4 Results

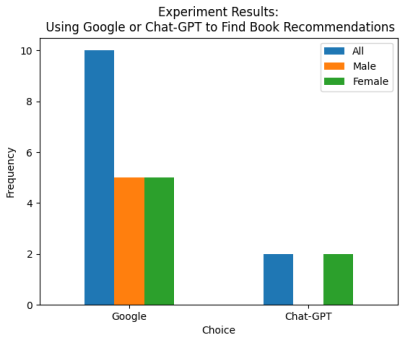
The results of the experimental part of the questionnaire regarding the choice of using ChatGPT in different scenarios are presented in Figure 2. For the search and recommendation task, 10 out of 12 chose Google to solve the problem. In contrast, 10 out of 12 preferred using ChatGPT for the task of learning and understanding maths problems. As for the last experimental question, 8 out of 12 chose ChatGPT as a tool to write a cover letter.

The population was divided into 2 groups depending on reported gender, to conduct a one-way ANOVA analysis with gender as the independent variable. ANOVA was applied to each item in the questionnaire to determine if there was any difference in responses between men and women, with a significance level of $\alpha = 0.05$. The null hypothesis in the test is the assumption that there's no difference between the groups. With an $\alpha = 0.05$, the null hypothesis was rejected for each item in the questionnaire with a $p - value < 0.05$. An excerpt of the ANOVA results is displayed in Table 3, showing 8 items with the highest p-values and 7 items with the lowest. Each of the 7 lowest had a $p - value < \alpha = 0.05$. A pairwise Tukey test was conducted that confirmed the results from ANOVA, see Table 4.

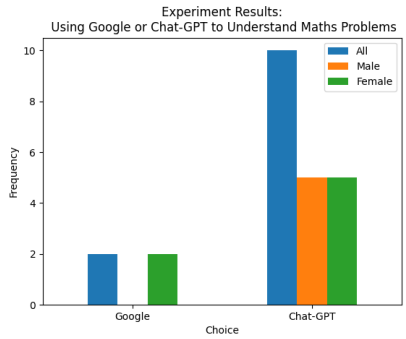
The table shows the mean difference between the groups for each item, the adjusted p-values, lower and upper confidence level bounds, whether the null hypothesis can be rejected, and which group had the highest mean value. The results show that there is a significant difference for the items:

- "AI does not scare me at all"
- "AI applications like ChatGPT makes me feel uneasy"
- "On average, how much do you currently use ChatGPT?"
- "I trust that the results obtained by using ChatGPT are accurate"
- "ChatGPT is not compatible with other systems I use"
- Familiarity with the experiment topic of writing cover letters
- Familiarity with the experiment topic of Sci-Fi litterature

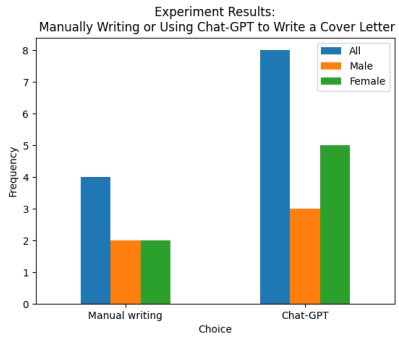
Two questions that controlled for familiarity with a topic in relation to ChatGPT use in the experiment had a significant difference with gender as an independent variable, as shown in Figure 4. The group of female respondents had a



(a) Method preference for finding book recommendation problem



(b) Method preference for understanding maths problems



(c) Method preference for writing a cover letter

Fig. 2: Resuts of the experimental part of the questionnaire, showing the distribution of answers for the female, male, and combined populations

Table 3: Excerpt from ANOVA results with gender as independent variable from the collected data including results with the highest and lowest p-values and their corresponding TAM variable. Items belonging to the experiment are left empty in the TAM variable column.

Statement	**p-value**	**TAM Variable**
My supervisors or instructors do not mandate me to use ChatGPT	0.9544	VOL
AI applications like ChatGPT makes me feel uncomfortable	0.9229	CANX
Level of experience in the service industry	0.8547	
I could effectively use ChatGPT even if there were no one available to guide me while using it	0.8154	CSELF
I feel capable of using ChatGPT for various tasks if I have experience using similar technology for similar tasks	0.8154	CSELF
I am confident that I can navigate and use ChatGPT after receiving initial guidance or instructions	0.8154	CSELF
I find ChatGPT to be easy to use	0.7114	PEOU
Cover letter scenario	0.7114	
AI does not scare me at all	0.0402	CANX
AI applications like ChatGPT makes me feel uneasy	0.0402	CANX
Level of experience with writing cover letters	0.0395	
On average, how much do you currently use ChatGPT?	0.0238	USE
I trust that the results obtained by using ChatGPT are accurate	0.0137	OUT
Knowledge about sci-fi or fantasy in general	0.0108	
ChatGPT is not compatible with other systems I use	0.0033	PEC

Table 4: Post hoc results using the Tukey test on the significant results obtained by ANOVA, at a 0.05 significance level

Statement	**Mean Diff**	**p-adj**	**Lower**	**Upper**	**Reject**	**Group with highest mean**
Knowledge about sci-fi or fantasy in general	1.2286	0.0108	0.352	2.1052	True	M
Level of experience with writing cover letters	-0.9714	0.0395	-1.886	-0.0568	True	F
On average, how much do you currently use ChatGPT?	0.9429	0.0238	0.1539	1.7318	True	M
AI does not scare me at all	1.1429	0.0402	0.0625	2.2232	True	M
AI applications like ChatGPT makes me feel uneasy	-1.1429	0.0402	-2.2232	-0.0625	True	F
I trust that the results obtained by using Chat-GPT are accurate	1.0571	0.0137	0.2682	1.8461	True	M
ChatGPT is not compatible with other systems I use	-1.7429	0.0033	-2.7582	-0.7275	True	F

F = Female, M = Male

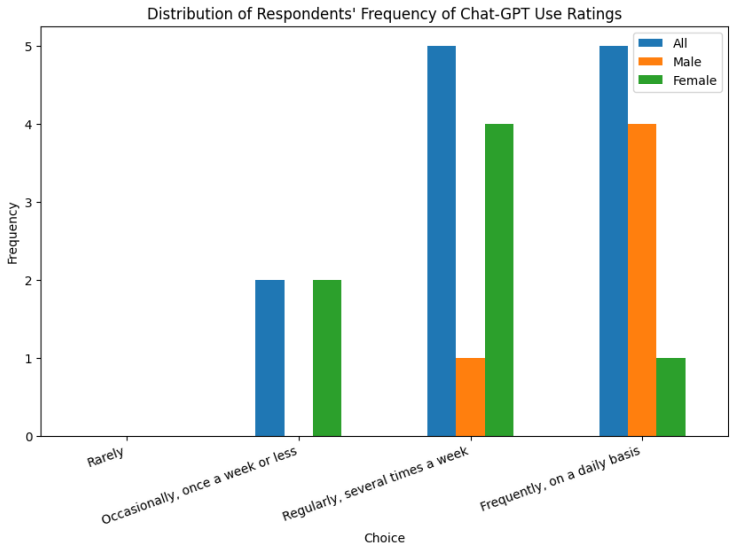
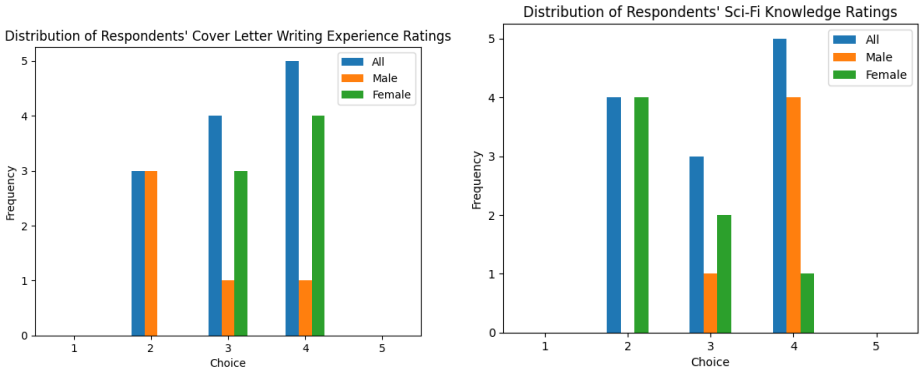


Fig. 3: Difference in distribution on current ChatGPT use frequency

higher mean regarding reported cover letter experience, and the group of male respondents had a higher mean in reported Sci-fi knowledge. However, the corresponding experiment tasks, writing a cover letter and finding book recommendations (as seen on Figure 2), did not indicate any difference in the ANOVA test.

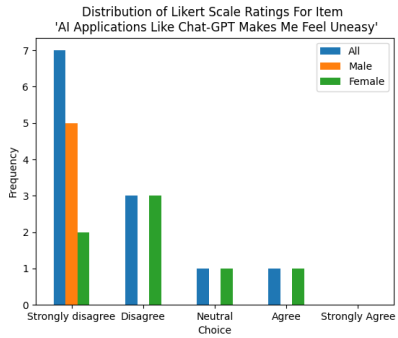


(a) Cover Letter Writing Experience (b) Sci-Fi Knowledge

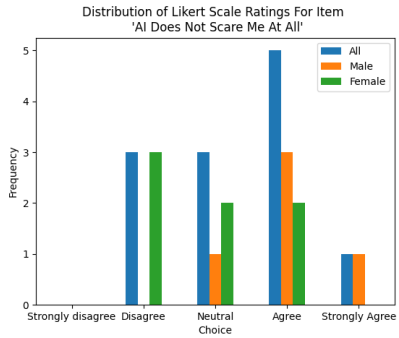
Fig. 4: Responses for questions regarding familiarity with the topics in the experimental part of the questionnaire, where 1 represents No knowledge/experience, and 5 represents advanced knowledge/experience. The figures show the distributions of female, male and combined populations' responses for the Sci-Fi and cover letter topics which had a significant difference according to ANOVA and Tukey analysis.

From the ANOVA results, there is a difference in the current use frequency between the male and female respondents, as displayed in Figure 3, where the male participants reported a more frequent use. However, when it comes to the experimental results of ChatGPT vs non-ChatGPT use, there was no significant difference in any of the scenarios. Furthermore, the TAM3 model includes three more items for analyzing Behavioral Intent of Use, in which no difference was detected: "I intend on keep using, or start using, ChatGPT", "I predict that I would keep using, or start using, ChatGPT", and "I plan on keep using, or start using, ChatGPT in the next 3 months".

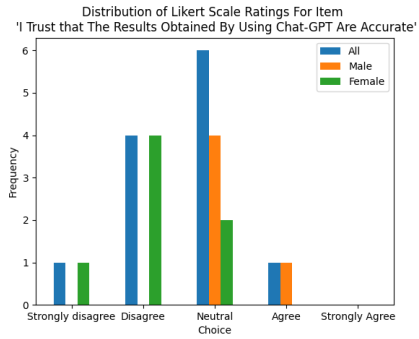
Three Likert Scale items from TAM3 and one added by the author had a significant difference according to ANOVA and the Tukey test, see Figure 5. The items in figures 5a and 5b are two out of four items that measure AI anxiety according to TAM3. While the participants overall disagreed with the statement "AI Applications Like ChatGPT Makes Me Feel Uneasy", where one participant remained neutral and another agreed, the male participants had a lower mean, thus stronger disagreement with the statement. For the statement "AI does not scare me at all" the responses ranged from Disagree to Strongly Agree, where the male participants reported a higher mean, thus displaying stronger agreement. These results indicate lower levels of AI anxiety amongst the male participants.



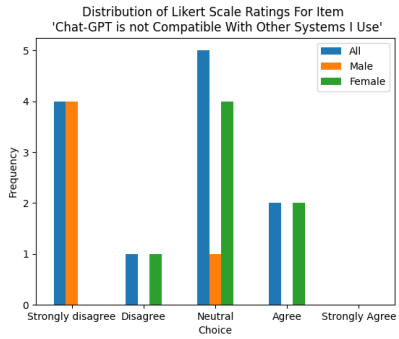
(a) Likert scale responses for the question "AI Applications Like ChatGPT Makes Me Feel Uneasy"



(b) Likert scale responses for the question "AI Does Not Scare Me At All"



(c) Likert scale responses for the question "I Trust That The Results Obtained By Using ChatGPT Are Accurate"



(d) Likert scale results for the question "ChatGPT is Not Compatible With Other Systems I Use"

Fig. 5: Distribution of the Likert Scale items derived from TAM3 that had significantly different means according to ANOVA and pairwise Tukey Test

The item in Figure 5c, "I trust that the results obtained by using ChatGPT are accurate", aims to measure the perceived output quality of ChatGPT, motivated by the hallucination effect and biases in LLMs. The group of female participants had a lower mean in this item indicating lower trust in the results obtained by ChatGPT, whereas the male participants had a more neutral perception, with only one participant agreeing with the statement. This is an interesting juxtaposition to another item "The quality of the results I obtain from ChatGPT is high" which had no significant difference between the groups and where 8 out of 12 participants agreed with the statement.

The male participants had a lower mean for the last item in Figure 5d, "ChatGPT is not compatible with Other systems I use", hence displaying stronger disagreement with the statement. This shows that the male participants perceive ChatGPT to be compatible with the systems they use.

5 Discussion

Out of the 61 items analyzed using ANOVA and Tukey test, only 6 items had a significant difference in means with gender as an independent variable. The only significant difference in use was the current frequency of use, where the male participants had a higher mean. No difference was found in the experiment or the other items that also measure future intent to use. This difference could perhaps be explained by the observation that the male participants perceive ChatGPT to be compatible with other systems they use at a higher rate, thus it may be easier to integrate into their current habits. In two out of four items measuring AI anxiety, 'AI Applications Like ChatGPT Makes Me Feel Uneasy" and "AI does not scare me at all", although the AI anxiety indicators were generally low, the male participants reported lower levels than the female participants. This observation could potentially explain the difference in current ChatGPT use.

The findings regarding the output quality items are interesting. While 8 out of 12 participants agreed with the statement "The quality of the results I obtain from ChatGPT is high", the ratings for the statement "I trust that the results obtained by using ChatGPT are accurate" were skewed towards the Neutral and Disagree side of the scale. This suggests that individuals may have an overall positive perception of the quality of the results, while still holding reservations towards the results' accuracy. The findings of how these seemingly contradicting thoughts can co-exist in the context of AI usage could be an interesting avenue for future research.

Given that the study population is characterized by high levels of technology knowledge and experience, the discovery of differences in AI anxiety items is intriguing. According to TAM3, experience is theorized to have a moderating effect on computer anxiety [18] which makes this difference surprising. Moreover, a study identified a positive correlation between exposure to Sci-fi media and AI fear [12], a finding contradicted by the results of this study, as the male participants reported higher ratings for Sci-fi knowledge and lower levels of AI

anxiety. This suggests that there could be other factors that may be influencing these outcomes.

There are some notable limitations to the study that may impact the results. First and foremost, due to the small sample size of $N = 12$, the results can be highly impacted by individual differences rather than general patterns. Secondly, due to the homogeneous target group, this paper only gathers insights into the acceptance of ChatGPT amongst students in Interaction Technology and Design in Sweden. This group, with a high level of experience and knowledge about technology, is not representative of the general public. The limited sample size and homogeneity regarding technical experience within the sample population constrain the generalizability of the study results. Thirdly, only ANOVA and the Tukey test were used in this study's statistical analysis. Considering that TAM3 is a rather complex model, further research may use more sophisticated and complex analysis methods to evaluate the results. Thus, more research is needed to verify and understand the findings of this study.

References

1. Sergio Luis Nánez Alonso, Javier Jorge-Vázquez, Pablo Arroyo Rodriguez, and Beatriz Maria Sastre Hernández. Gender gap in the ownership and use of cryptocurrencies: Empirical evidence from Spain. *Journal of Open Innovation: Technology, Market, and Complexity*, 9(3):100–103, September 2023.
2. Francisco G Barbeite and Elizabeth M Weiss. Computer self-efficacy and anxiety scales for an Internet sample: testing measurement equivalence of existing measures and development of new scales. *Computers in Human Behavior*, 20(1):1–15, 2004.
3. Erik Brynjolfsson, Danielle Li, and Lindsey R Raymond. Generative AI at work. Working Paper 31161, National Bureau of Economic Research, April 2023.
4. Fred D. Davis. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly*, 13(3):319–340, 1989. Publisher: Management Information Systems Research Center, University of Minnesota.
5. Berkeley J. Dietvorst, Joseph P. Simmons, and Cade Massey. Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, 144(1):114–126, 2015. Place: US Publisher: American Psychological Association.
6. Berkeley J. Dietvorst, Joseph P. Simmons, and Cade Massey. Overcoming Algorithm Aversion: People Will Use Imperfect Algorithms If They Can (Even Slightly) Modify Them. *Management Science*, 64(3):1155–1170, March 2018. Publisher: INFORMS.
7. David Gilbert, Liz Lee-Kelley, and Maya Barton. Technophobia, gender influences and consumer decision-making for technology-related products. *European Journal of Innovation Management*, 6(4):253–263, December 2003.
8. Nicole Gross. What ChatGPT Tells Us about Gender: A Cautionary Tale about Performativity and Gender Biases in AI. *Social Sciences*, 12(8):435, August 2023.
9. Abid Haleem, Mohd Javaid, and Ravi Pratap Singh. An era of ChatGPT as a significant futuristic support tool: A study on features, abilities, and challenges. *BenchCouncil Transactions on Benchmarks, Standards and Evaluations*, 2(4):100089, October 2022.

10. Ruchika Jain, Naval Garg, and Shikha N. Khera. Adoption of AI-Enabled Tools in Social Development Organizations in India: An Extension of UTAUT Model. *Frontiers in Psychology*, 13:893691, June 2022.
11. Ekaterina Jussupow, Izak Benbasat, and Armin Heinzl. Why Are We Averse Towards Algorithms? : A Comprehensive Literature Review on Algorithm Aversion. *ECIS 2020 Research Papers*, June 2020.
12. Yuhua Liang and Seungcheol Austin Lee. Fear of Autonomous Robots and Artificial Intelligence: Evidence from National Representative Data with Probability Sampling. *International Journal of Social Robotics*, 9(3):379–384, June 2017.
13. Katie Paul, Akash Sriram, and Akash Sriram. Meta's Twitter rival Threads surges to 100 million users faster than ChatGPT. *Reuters*, July 2023.
14. Partha Pratim Ray. ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems*, 3:121–154, 2023.
15. Anna Lena Schulte Steinberg and Christoph Hohenberger. Can AI close the gender gap in the job market? Individuals' preferences for AI evaluations. *Computers in Human Behavior Reports*, 10, May 2023.
16. Yiqiu Shen, Laura Heacock, Jonathan Elias, Keith D. Hentel, Beatriu Reig, George Shih, and Linda Moy. ChatGPT and Other Large Language Models Are Double-edged Swords. *Radiology*, 307(2):e230163, April 2023. Publisher: Radiological Society of North America.
17. UNESCO, OECD, IDB, Clementine Collett, Gina Neff, and Livia Gouvea Gomes. *The Effects of AI on the Working Lives of Women*. United Nations Educational, Scientific and Cultural Organization, Paris, 2022. OCLC: 1338242553.
18. Viswanath Venkatesh and Hillol Bala. Technology Acceptance Model 3 and a Research Agenda on Interventions. *Decision Sciences*, 39(2):273–315, May 2008.
19. Viswanath Venkatesh and Fred D. Davis. A Theoretical Extension of the Technology Acceptance Model: Four Longitudinal Field Studies. *Management Science*, 46(2):186–204, February 2000.

A comparison study between RGB Camera based mapping and LiDAR based mapping algorithms using ROS and Gazebo

Anthony Mallma

Department of Computing Science
Umeå University, Sweden
amallmav@uni.pe

Abstract. The extensive research done in Simultaneous Localization and Mapping (SLAM) has made it possible for different approaches to the topic, such as Karto-SLAM, Hector-SLAM, Gmapping, and RTAB-Map, to be available as open-source algorithms for testing in different applications. In this work, we use the ROS (Robot Operating System) implementation of those four approaches to map a custom indoor environment designed in Gazebo. The experiments are done in simulation by teleoperating the Turtlebot3 through the same trajectory within the environment each time, and the four resulting maps are compared to the designed ground truth map to determine which method outputs better mapping accuracy. The experimental results give evidence about what algorithms to consider as the first option to perform a mapping task for an indoor environment.

1 Introduction

Within the mobile robotics field, one of the main goals is making a robot navigate autonomously in a real-world environment similar as we humans can. In this matter, the localization and mapping problems are milestones to achieve that goal. The difficulty of these problems is that most real-world applications require the robot to explore unknown environments. Then the robot will be required to build a map, this means discovering the positions of the landmarks in the environment, but to get these positions it is necessary to know the location of the robot. Moreover, it is not possible to know the location of the robot without a map. Thus, both the localization and mapping problems need to be addressed simultaneously, which is the objective of the study of the SLAM (simultaneous localization and mapping) technique. In this paper we compare Karto-SLAM, Hector-SLAM, and Gmapping, which are different SLAM techniques based on LiDAR, with RTAB-Map SLAM (the technique that uses RGB-D camera), to determine which outputs better mapping accuracy when performing in an indoor environment. The accuracy of the maps is obtained using the Structural Similarity Index (SSIM) using a ground truth map as a reference.

Suna Bensch (Eds.): Umeå's 27$^{\text{th}}$ Student Conference in Computing Science USCCS 2024, pp. 73–85, January 2024.

2 Background

This work relies on the basic theory of mobile robotics, particularly in knowledge about the main sensors, algorithms, and simulation tools available in the field. For instance, regarding the sensors used in mobile robots, it is important to have an understanding of LiDAR and RGB Cameras. The term LiDAR is the acronym for Light Detection and Ranging. This sensor emits laser beams and uses the time of flight technology to detect objects in the surroundings and the distance to these objects. Conversely, an RGB Camera is a type of camera that captures the red, green, and blue components of light to output colored pictures that are conformed by pixels that could be processed by Image processing software.

In addition to sensors, it is crucial to know the frameworks where the algorithms and simulation are developed. This paper uses ROS, known as Robot Operating System, which is a framework built especially for robotics applications. The main advantage is that provides a standard for robotics software that developers can use and reuse in any robot. Furthermore, within ROS, different built-in packages can be installed, such as Rviz which is a 3D visualization tool that shows what is happening with the ROS topics in a 3D representation, it can also be used as a debugging tool. On the other hand, when a real robot is not available, a simulation tool (compatible with ROS) called Gazebo can be used. It simulates the gravity, physical properties, and hardware control of the robot. It is a useful tool to replace the real robot, such as the Turtlebot3. This is a popular open-source robot used in research. There is an available model in Gazebo that has models of the actual sensors the real-world robot uses to interact with the environment.

Finally, the main algorithms needed for this research are SLAM and SSIM. SLAM stands for Simultaneous Localization and Mapping. It studies the problem of making a body navigate in a previously unknown environment while constantly building and updating a map of its workspace using only onboard sensors. This technique becomes crucial when a robot must be truly autonomous, when there is no prior knowledge of the environment, and when we can not use external positioning systems. Additionally, the other important algorithm is SSIM which stands for Structural Similarity Index [4], and is used as a technique for comparison of images. It considers a testing image and a reference image as inputs and then it outputs a number between 0 and 1, which indicates the geometrical distortion found by computing the displacement in the position of the pixels. The closest to 1 means that less geometric distortion, discontinuities, and misalignments were found.

3 Earlier Work

This research relies on knowledge in the following main areas: robot and environment simulation, mobile robotics sensors simulations, and SLAM. Consequently, this section explores the relevant literature related to those main areas.

First of all, the use of simulations for early-stage problems or educational matters has been popular since it allows the gathering of data and results that

approximate real-world behavior. In the field of robotics, Gazebo and ROS are widely used frameworks to solve different applications.

The paper [2] uses these frameworks to create a simulation for a real crawler mobile robot by modeling the components of the robot, by adding physical and collision properties in the simulation. In another work, [4] shows the usage of a simulated LiDAR sensor and a Camera to perform fruit mapping on a simulated agricultural area, confirming that Gazebo and ROS have packages to replicate different types of real sensors in the simulation. Having a model of a robot is not enough to perform tasks in an environment, it needs control and autonomy capabilities. One of the main features wanted in mobile robotics is the capability to navigate in an unknown environment. This issue is aimed to be solved by the SLAM technique which estimates the poses of the robot and the map at the same time.

There is wide research on SLAM that led to the development of multiple types of it, which are based on different underlying methods. For instance, Karto-SLAM [7] is a graph-based SLAM method that aims to solve the large pose graphs optimization problem by using Sparse Pose Adjustment (SPA). SPA shown to outperform other optimization methods, such as Stochastic gradient descent and Decomposed nonlinear systems, in terms of accuracy and convergence speed, thus reducing computation time.

Additionally, there are particle filter-based methods [10], which consider the probability distributions, used to represent the uncertainty, to be non-gaussian. In these methods, a sample of equally weighted particles (the particles are possible estimates of the robot state) is spawned over the map to capture the current probability distribution. As the robot moves and observations are made, some particles will increase their weight based on how close their predicted observation to the actual observation (obtained with the sensor) was, then based on the weights the particles get resampled, the robot moves again and the same process continues. Gmapping [3] is a particular particle filter-based method that uses the Rao-Blackwellization factorization to compute first the probability distribution for the robot's poses and then given that information, computes the distribution for the map.

Moreover, Hector-SLAM [6] is a method that does not depend on odometry and was developed specially to map unknown environments in Urban Search and Rescue (USAR) missions. Those are unstructured environments that will require us to estimate the 6 degrees of freedom in the robot's movement. This is done using an IMU and to correct the position drift, due to sensor noise, a scan matcher is used to get the best alignment of the laser scans given the current map [6]. One big advantage of these SLAM methods is that they have a ROS implementation ready to be used. This is confirmed in the paper [11] that has studied Gmapping, Hector SLAM, and Karto SLAM which are LiDAR-based 2D SLAM that he implemented using Gazebo and ROS, his experiments show that high precision maps can be obtained using those algorithms. Thus, showing that SLAM techniques are implementable and available as packages in those frameworks.

On the other hand, performing SLAM using an RGB camera is a milestone of this paper, and the next paragraphs review existing literature specifically on this topic.

In the research [9] a Turtlebot robot that includes the Raspberry Pi camera is used on the ROS framework, the work shows that this camera can be configured in simulation and outputs reliable data making it a good option for camera-based SLAM. Therefore, confirming that it is possible to set up these types of cameras in Gazebo through simulation. The work in [1] studies a set of monocular SLAM methods, such as ORB-SLAM, that by default reconstruct a 3D map. However, its performance is not reliable enough to detect office-style walls colored in light paints, thus leaving gaps in the map that can lead to collisions.

Furthermore, RTAB-Map is another camera-based method that works with pose graphs that only consider the locations more likely to cause loop closure detection. This is done to limit the size of the graph so the method can operate for longer periods and in large environments without surpassing a threshold processing time. The method uses an RGB camera which is synchronized with the estimated pose of the robot obtained from an odometry module, this information is used in the graph optimization step that outputs the map of the environment, [8]. One advantage of RTAB-map is the availability of a straightforward implementation in ROS that is stable and can output a 2D map. For example, publications such as [5] mention the existence of a ROS package RTAB-Map ros-pkg, that outputs a 3D point cloud map and a 2D grid map. The 2D grid map is the same type we obtain mapping with the LiDAR sensor, then is suitable to compare with the LiDAR-based maps. It also gives the idea that to make the comparison feasible we must rely on a ground truth map.

The literature reviewed gives a picture of the options available to perform lidar-based SLAM and camera-based SLAM. This work selects from those options, the ones showing evidence of successful and stable implementation in ROS and Gazebo. Thus, this paper builds upon that previous work and contributes by using a reliable simulation of an environment and a robot that will execute four SLAM methods implemented in ROS, to finally make a comparison study of the map output by those methods, more specific details are shown in the next section.

4 Method

In this work, we will generate a map for an indoor environment four times, each time a different SLAM technique will be used. The first three maps will be generated using Karto-SLAM, Hector-SLAM, and Gmapping, all of these are LiDAR-based algorithms that output a 2D occupancy map. For the generation of the fourth map, RTAB-Map SLAM is used. This last algorithm is the only one that uses an RGB camera and its resulting map will define if the addition of the camera makes a difference when mapping a virtual indoor world in ROS and Gazebo. For all the cases the procedure follows this order: simulating a robot with the sensor, simulating the environment, and performing the mapping task.

4.1 Pure LiDAR-based approaches

The robot chosen for this work is the TurtleBot3 Waffle, which is the most popular open-source robot for research. It has a LiDAR model LDS-01 which is a 2D laser scanner that collects data 360 degrees around the robot. This robot has an available package for ROS and Gazebo, which is used to get the TurtleBot3 Waffle simulation in Gazebo. Within this package, a simulation of the LiDAR sensor is included, and it can provide an array that contains the distances to different objects hit by the laser beams.

Additionally, a simulation of an indoor environment is needed. This simulation can be made using Gazebo and the advantage is that a custom environment can be built by using the Gazebo drawing tools. For this paper, a simple office-like environment was designed that is composed of walls and open doors, as can be seen in Figure 1. The layout of this environment is shown in Figure 2, and this is the ground truth map that will be used as a reference image for the later comparison.

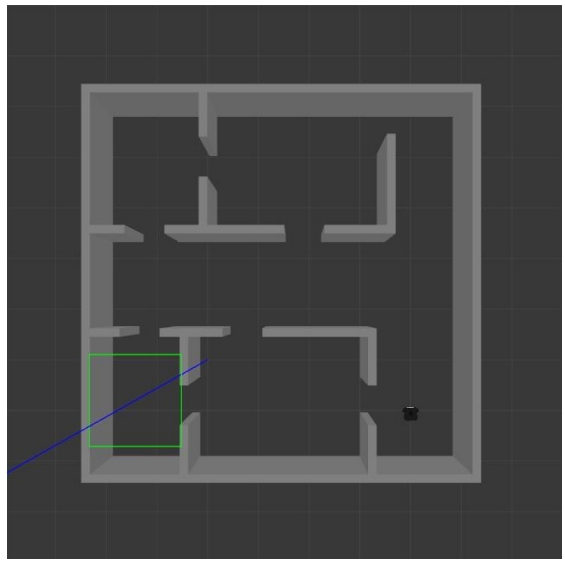


Fig. 1. Custom virtual indoor environment to be mapped.

Having the simulation of the robot and the environment available, the next step is to get its map. The first method used to generate the map was Karto-SLAM. It is part of the SLAM package available for the Turtlebot3 in ROS. When this method is executed, it runs Rviz and shows the current state of the map and the objects detected by the LiDAR, see Figure 3.

The next step is to move the robot within the simulated environment, to do this the teleoperation feature of the TurtleBot3 is run, and then using the keyword the robot moves through the environment following the path shown

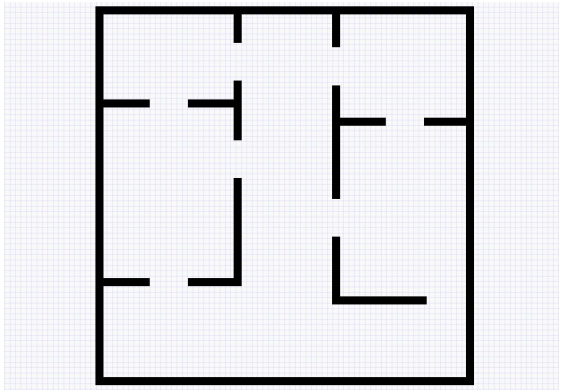


Fig. 2. Layout of the virtual world to be mapped.

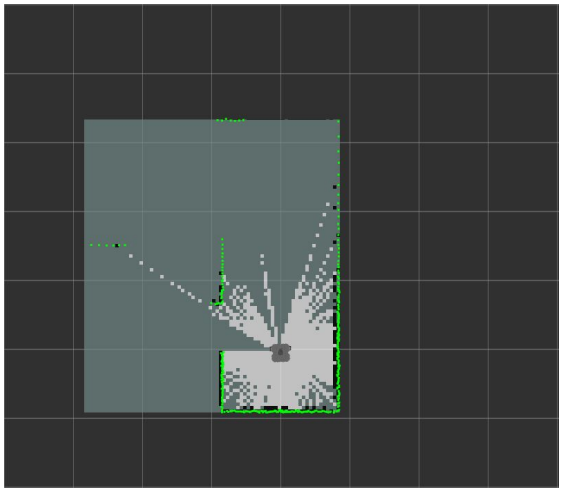


Fig. 3. Scan detections when initializing Karto-SLAM.

in Figure 4. The path ensures the robot explores the whole environment. Once finished, the final map can be seen in Rviz and is saved as a pgm file.

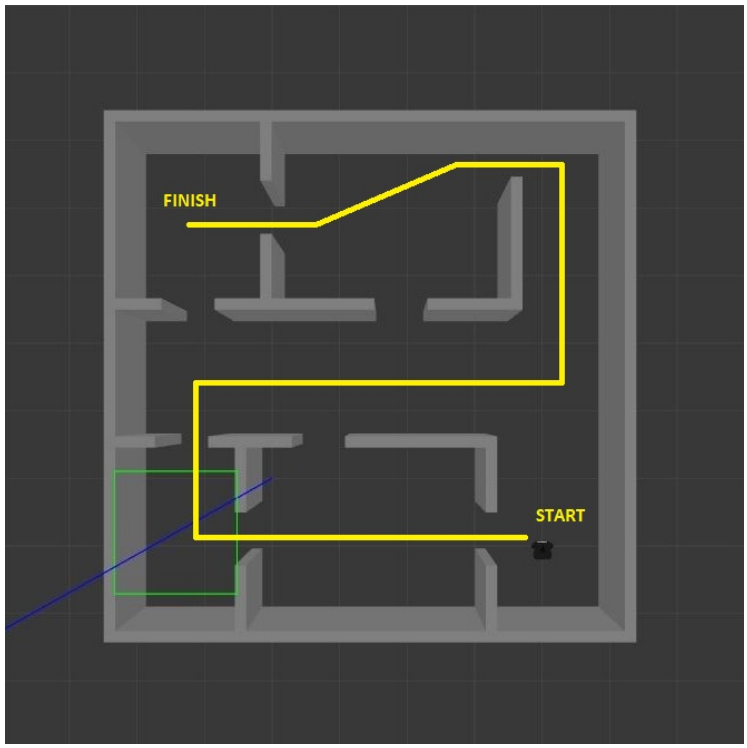


Fig. 4. Path followed by the Turtlebot to map the environment.

The same procedure is used to generate the maps with the rest of the methods: Hector-SLAM and Gmapping, which are included in the same SLAM package. The initial scans when the algorithms are initialized can be seen in Figure 5. To make a fair comparison in all cases the robot followed the same path, therefore the mapping quality is the result of the algorithms themselves. Finally, the maps obtained will be shown in the results section.

4.2 RGB Camera-based approach

The TurtleBot3 Waffle includes an Intel RealSense camera R200, which is RGB. The Gazebo simulation of this camera runs automatically when we run the simulation of Turtlebot3. This paper considers the RTAB-Map method to map the environment since it supports RGB camera SLAM, also there is an available package in ROS to run this method. Similar to the cases in the previous subsection, the algorithm open Rviz and show the state of the map and the objects detected by the sensors, see Figure 6.

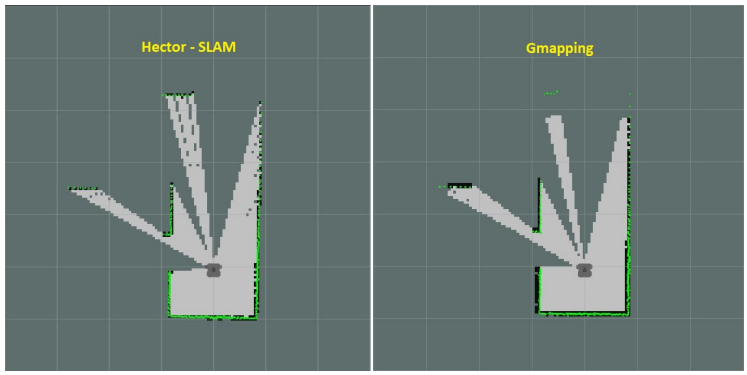


Fig. 5. Scan detections when initializing Hector-SLAM and Gmapping respectively.

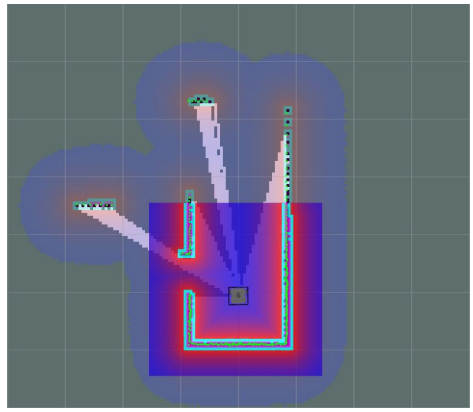


Fig. 6. Scan detections when initializing RTAB-Map.

Again teleoperation is used, following the path in Figure 4, to map the environment. The result is a 2D occupancy grid map that is comparable to the maps obtained in the LiDAR-based cases.

4.3 Analyzing the Maps obtained

In this work, an office-like environment was designed in Gazebo, see Figure 1. Mapping this environment in an ideal world where there are no uncertainties at all, would output a map that looks the same as the layout in Figure 2. However, even when working with a simulated robot and simulated sensors, which certainly have less uncertainty than the real ones, the maps obtained will differ from the layout. The analysis in this work is based on how much difference exists between the maps obtained from the different SLAM methods and the reference image (layout).

Since the comparison of the maps is reduced to a problem of comparing the similarity of one image to a reference image, this paper uses the Structural Similarity Index (SSIM) technique which returns a numerical value that represents the visible difference between the compared images. A higher value indicates that the structure of the tested image is more similar to the reference image. This method was chosen because it considers geometrical distortion and misalignment in the images. Therefore, it will detect misaligned walls and shifted or discontinued pixels in the walls, which are the most common situations that will be present in the resulting maps.

5 Results

Having all the maps completed in Rviz, they were saved as pgm files, and in order to visualize them MATLAB was used. MATLAB can read those files and plot them as figures. For each SLAM method, the map was plotted and the resulting maps are shown in Figure 7.

In order to compare the maps to the reference, it was required to ensure they have the same size and resolution, this was fixed by checking the size of the pgm images in MATLAB and scaling all the images to the same size. The resulting common size of the images is 704x704 pixels. For the case of the ground truth image its background was turned to white so it matches the background in the maps, and the result is shown in Figure 8.

With all images having the same size, the SSIM technique is used. There is a function available for this in MATLAB, it requires to input both the reference image and the tested image. Each of the maps from the different methods is input to this function together with the reference image, obtaining the results shown in Table 1.

6 Discussion

This paper performed a comparative analysis of different maps generated by using four SLAM methods available in ROS. The methods were tested using a

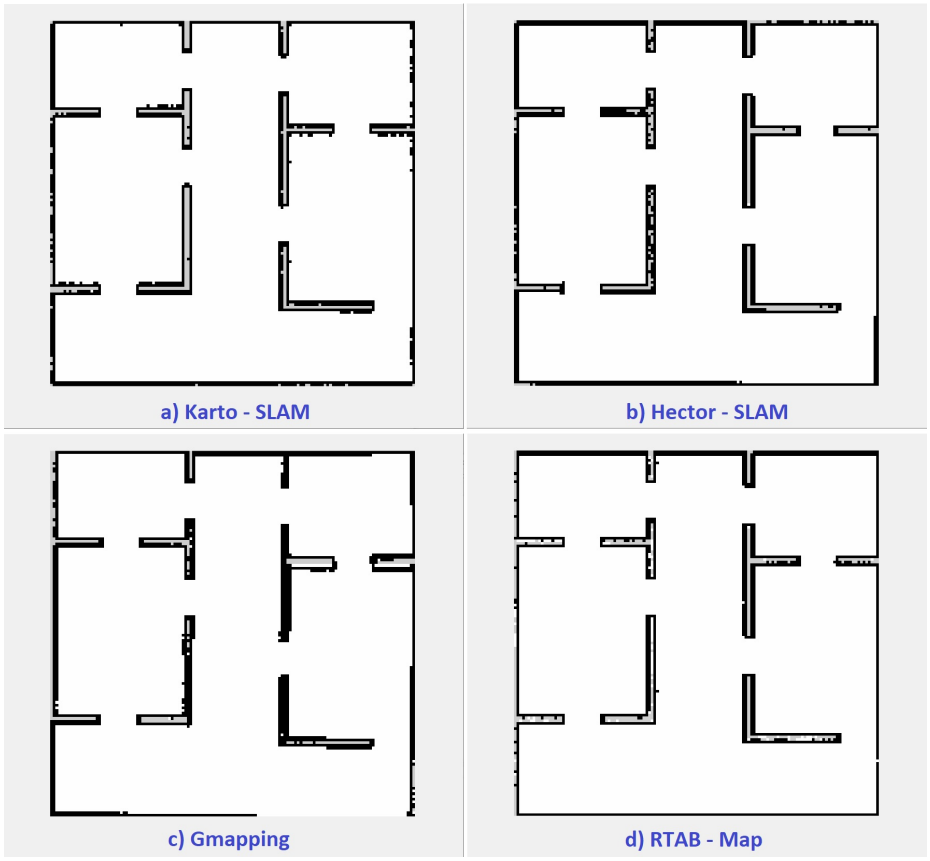


Fig. 7. Different maps obtained when running the SLAM algorithms and navigating the environment.

SLAM Technique	**SSIM Result**
Karto - SLAM	0.8097
Hector - SLAM	0.8113
Gmapping	0.8151
RTAB - Map	0.8039

Table 1. SSIM results when comparing the maps with the ground truth.

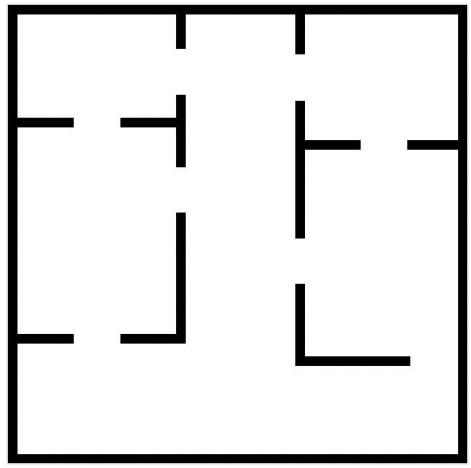


Fig. 8. Ground truth map of the environment.

simulated robot in a virtual environment and to make the comparison fair the robot went through a common trajectory in all cases.

To avoid any subjective interpretations the comparison is done by using a number that indicates how much similarity each resulting map has to a reference map. This number is known as the SSIM value and from the results it is concluded that the mapping accuracy obtained goes from the highest to the lowest value in the following order: Gmapping (SSIM = 0.8151), Hector-SLAM (SSIM = 0.8113), Karto-SLAM (SSIM = 0.8097), and RTAB-Map (SSIM = 0.8039).

These results can be explained by focusing on the key characteristics of each SLAM method. In the case of Gmapping, it has proved to satisfactorily work in planar and office-like environments when the mobile robot has a slow movement, since it helps to make the odometry estimation more accurate. Conversely, Hector-SLAM is a method that shows its strengths in uneven and unstructured terrain, and it exploits the modern LiDAR's high update rate when the robot moves fast, such performance improves in smaller scenarios.

Then, we have Karto-SLAM and RTAB-Map which belong to the family of Graph-based SLAM methods. The GraphSLAM method's strength is that they don't require notable onboard processing capability due to the graph optimization performed. For instance, Karto-SLAM's goal is to optimize large pose graphs and reduce the computation time needed. In consequence, making relevant improvements in the mapping accuracy is not its strong point. Moreover, regarding RTAB-Map, it is a technique intended to operate in large environments by limiting the number of locations for loop closure detection. This reduces the size of the graph and the processing time. Since this approach works with an RGB camera, its performance improves as more distinct features are available in the environment to make the feature match step during loop closure detection. The added value of RTAB-Map is that outputs a 3D map in addition to the 2D map that all the other methods output as well. However, in this case, this does not

contribute to the performance of RTAB-Map since the comparison is made only using 2D maps.

Now, the mentioned strengths of the algorithms must be analyzed in context. The context is the characteristics of the ground truth environment designed for this paper.

First, the designed environment is small and does not have long corridors, thus the main advantages of GraphSLAM methods are not significant in this context since no large pose graphs will be present. For that reason and the fact that this paper is evaluating mapping accuracy, Karto-SLAM, and RTAB-Map perform a bit worse than Gmapping and Hector-SLAM. Second, this environment is composed only of walls and corners and apart from that does not have any highly distinctive features or landmarks. Therefore, one of the strengths of RTAB-Map is downplayed, and for that reason, it performs worse than the other algorithms. In comparison to Karto-SLAM, when generating a 2D-Map, a LiDAR (used by Karto-SLAM) has more field of view than a camera, thus working with LiDAR results in low-drift navigation which makes the map more accurate, this explains why Karto-SLAM mapping accuracy is a bit better than RTAB-Map.

Third, for the comparison between Gmapping and Hector-SLAM, we have that the environment is not uneven and the speed of the Turtlebot3 is slow (maximum 0.22 m/s). Therefore, the two main strengths of Hector-SLAM have been lessened. Moreover, Gmapping benefits from this environment which is planar and similar to an office, which explains why it performs better than Hector-SLAM.

In conclusion, given a small indoor, office-like environment, Gmapping has proved to be the SLAM algorithm that outputs the more accurate map. Thus, when trying to create a map of an office in the real world, Gmapping is a simple algorithm worth trying as an initial solution for the task. In future work, the behavior of the studied algorithms can be tested in the real world, using a real Turtlebot3 that explores an office within the university campus. This will give an insight into how real-world uncertainties and disturbances make a difference in the mapping accuracy of the proposed SLAM methods.

References

1. Alexander Buyval, Ilya Afanasyev, and Evgeni Magid. Comparative analysis of ROS-based monocular SLAM methods for indoor navigation. In Antanas Verikas, Petia Radeva, Dmitry P. Nikolaev, Wei Zhang, and Jianhong Zhou, editors, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 10341 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, pages 305–310, March 2017.
2. Alexandra Dobrokvashina, Roman Lavrenov, Evgeni Magid, Yang Bai, and Mikhail. Svinin. How to create a new model of a mobile robot in ROS/Gazebo Environment: An extended tutorial. *International Journal of Mechanical Engineering and Robotics*, 12(4):192–199, 2023.

3. Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.
4. Novian Habibie, Aditya Murda Nugraha, Ahmad Zaki Anshori, M. Anwar Ma'sum, and Wisnu Jatmiko. Fruit mapping mobile robot on simulated agricultural area in Gazebo simulator using simultaneous localization and mapping (SLAM). In *2017 International Symposium on Micro-NanoMechatronics and Human Science (MHS)*, pages 1–7, 2017.
5. Ilmir Z. Ibragimov and Ilya M. Afanasyev. Comparison of ROS-based visual SLAM methods in homogeneous indoor environment. In *2017 14th Workshop on Positioning, Navigation and Communications (WPNC)*, pages 1–6, 2017.
6. Stefan Kohlbrecher, Oskar von Stryk, Johannes Meyer, and Uwe Klingauf. A flexible and scalable SLAM system with full 3D motion estimation. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, pages 155–160, 2011.
7. Kurt Konolige, Giorgio Grisetti, Rainer Kümmerle, Wolfram Burgard, Benson Limketkai, and Regis Vincent. Efficient Sparse Pose Adjustment for 2D mapping. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 22–29, 2010.
8. Mathieu Labbé and François Michaud. RTAB-Map as an open-source LiDAR and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics*, 36(2):416–446.
9. Marcus Nises. ROS-based implementation of a model car with a LiDAR and camera setup (dissertation). Master's thesis, Uppsala Universitet, 2023.
10. Yuhai Wei, Hui Zhang, Guang Deng, Hang Zhong, and Li Liu. Research on the SLAM of mobile robot based on particle filter method. In *2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, pages 1640–1645, 2019.
11. Xuexi Zhang, Guokun Lu, Genping Fu, Dongliang Xu, and Shiliu Liang. SLAM algorithm analysis of mobile robot based on LiDAR. In *2019 Chinese Control Conference (CCC)*, pages 4739–4745, 2019.

Compressing 3D Models Into Micro Meshes – A Usability Comparison Between Two Conversion Toolkits

Hendrik Otte

Department of Computing Science
Umeå University, Sweden
ens22hoe@cs.umu.se

Abstract. Micro meshes are a new way to store 3D models offering high geometric fidelity while consuming less memory and processing costs in raytracing applications. We analyze two toolkits that can convert traditional 3D models into the new compressed format. Both toolkits are compared in terms of usability, performance, compression, quality, and automation using a set of sample models. Experiments are conducted to identify strengths and weaknesses of both tools.

1 Introduction

How can we compute a two-dimensional picture given a mathematical description of a three-dimensional scene? This process of taking 3D scenes and convert them into 2D pictures is called rendering [9]. It is a major part of computer graphics and is used in computer games, animated movies, scientific simulations, and many more.

One building block for 3D rendering are 3D models (or 3D meshes). There are different ways to describe such a model. A common way is to define a finite set of vertices in a three-dimensional space where three vertices together can form a triangle. Multiple triangles build a surface and can be used to model complex shapes [9]. Even though the triangles are positioned in a 3D space, each triangle itself is a 2D object. Thus, many triangles are needed to approximate a very detailed and complex 3D geometry (see Fig. 1). Therefore, the amount of necessary memory to store the model can grow rapidly.

Given 3D models and additional parameters like textures, light sources, or materials, we can use a rendering algorithm to turn this description of a scene into a 2D image. The time and space needed for this computation increases with the complexity of the scene, i.e. the algorithm runs slower with more detailed models. Therefore, it has been of great interest to create more realistic images without increasing the complexity of a scene and its models. Techniques like Bump Maps [1], for example, emulate a more complex surface by changing the surface normals of an object. This is changing the way light interacts with it while keeping the actual geometry unchanged.

Suna Bensch (Eds.): Umeå's 27th Student Conference in Computing Science USCCS 2024, pp. 87–98, January 2024.

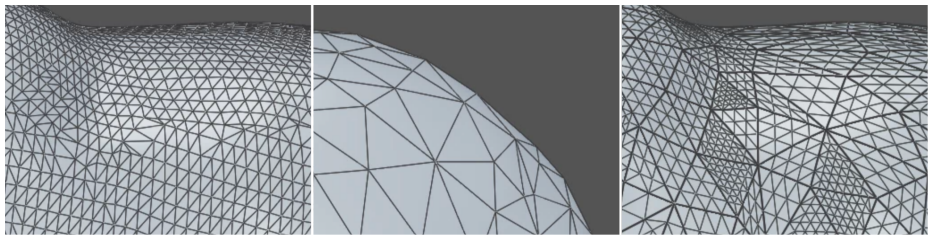


Fig. 1. Screenshot of original high resolution mesh (left) turned into low resolution base mesh (middle) with subdivided base triangles (right).

In the following, two pipelines are compared that aim to convert high resolution 3D meshes into the micro mesh file format. Micro meshes are a novel representation of 3D objects proposed by NVIDIA[1]. The goal of micro meshes is to compress complex 3D objects while preserving their high level of detail and at the same time offering native hardware support for decompression and ray tracing. While the general idea of compressing 3D meshed is not new [5, 3, 7], the micro mesh representation and its hardware-level support is. As of today, there are almost no tools for micro meshes available. We found two possible conversion pipelines that are designed for micro meshes. We compare these two possible pipelines while focusing on the quality but also on the usability in its current state and when integrated into modeling software for artists.

One pipeline is provided by NVIDIA[1]. It is meant to be integrated in higher level software but can also be used as a toolkit on its own. The second pipeline [8] combines various techniques from previous research but specifically tailors these to achieve high quality outputs. We will refer this second approach as MMT pipeline (based on the authors **M**aggiordomo, **M**oreton, and **T**arini). We try to answer the question: How does these two pipelines differ from each other in terms of various criteria, like quality, usability and performance. We aim to get an overview of the strengths and weaknesses of these tools.

To answer this question, both pipelines are used to compress a set 3D models into the new micro mesh format. A qualitative comparison of the output is used to identify differences in quality and the compression rates of both pipelines are compared together with their performance.

1.1 Micro Meshes

The general idea of micro meshes is to use a low detailed base mesh and then reapply the previous details by subdividing each triangle and then displacing these smaller triangles to approximate the original geometry[1] (see Figure 1). Only the base mesh with compressed displacement information are necessary to restore the details on the GPU (Graphics Processing Unit) during rendering.

The general way to compress a 3D model into the micro mesh format consists of three steps[1]: 1. generating the low resolution base mesh, 2. finding the dis-

[1] `https://www.nvidia.com/en-us/on-demand/session/gtcspring23-s51410/`

placement information to restore the details later, and 3. compressing the data into the new format.

During the first step, the high resolution input mesh is reduced to a mesh with less triangles. The resulting base mesh contains less information and details than the input mesh but it consumes less memory. To regain a high detailed surface, each base triangle is subdivided into 4^k smaller triangles for some positive integer k. Each triangle can have a different subdivision level k while neighboring base triangles can differ in subdivision level by at most one. Thus, areas with many details can be subdivided into smaller micro triangles while less complex areas can stay low resolution. These smaller micro triangles lay in the same plane as the base triangle and do not provide any additional details. Therefore each vertex of a micro triangle is displaced by a scalar factor along an interpolated normal direction. This subdivision and displacement is done during rendering on a GPU. Only the low resolution base mesh together with the displacement information are stored in a file and have to be loaded to the GPU. The micro triangle attributes are also stored in a coherent hierarchical storage order.

In addition to the lower file size, another benefit is that the subdivision level can be scaled up and down during rendering. If a 3D object, for example, takes up only a small area of the image, its resolution can be reduced because its high details cannot be shown anyway. When the object moves closer to the camera, its subdivision level can gradually scaled up to provide more details.

Is the micro mesh object used together with raytracing, usually an acceleration structure is build on all 3D objects/triangles in a scene to improve raytracing performance. With micro meshes, only the base mesh has to be added into the acceleration structure. Up and downscaling the subdivision level do not require rebuilding the acceleration structure. Full details can be found in the NVIDIA specifications[1].

1.2 NVIDIA Pipeline

The NVIDIA pipeline[1] takes a high resolution input mesh and reduces its complexity by iteratively removing edges and vertices using an edge collapse [4] technique. At each step there are many different options which collapse operation to perform. A cost function based on different attributes, like edge length or curvature, is used to choose the collapse operation with the lowest cost. This iterative process is not well suited for parallel computing. Therefore, the original mesh is split into multiple sub-meshes that are simplified on its own using a highly parallel GPU algorithm. The edge collapse operations are performed until a user-defined threshold is reached.

The history of collapse operations is then used to determine the subdivision level k of each base triangle and the length of the base triangle displacement vectors. Finally, the bounding hull is minimized to fit the new mesh.

1.3 MMT Pipeline

The MMT pipeline [8] is using the same basic pipeline structure as the NVIDIA toolkit but introduces additional optimization steps and is trying to optimize the base mesh specifically for the needs of micro meshes.

Given a high resolution input mesh, the MMT pipeline first removes outliers. Then, the base mesh is generated. Because the quality of the future micro mesh depends on the base mesh, a lot of effort is put into the generation of the base mesh. To reduce the amount of triangles in the input mesh, existing coarsening techniques (e.g. edge collapse [4] and more) are applied iteratively. At each step, there are various different options which operation can be performed. Using different cost functions tailored to specific needs of micro meshes, the MMT pipeline simulates each of these options and then executes the operation that minimizes the cost functions. Due to the huge amount of options, in the beginning an heuristic approach is used. Later, a more precise evaluation is performed. These cost functions are also used to stop the coarsening process and to end the base mesh generation. Together with the base mesh generation, the displacement vectors are calculated with a metric that is based on the orthogonality of adjacent triangles.

For each triangle of the base mesh, a subdivision level has to be found. A higher subdivision level leads to more and smaller triangles. The MMT pipeline aims to roughly match the total number of micro triangles with the number of triangles in the original mesh. To determine which parts of the base mesh should be more detailed than others, two main approaches can be used. A uniform and an adaptive approach. The uniform approach is based on the area covered by a base triangle. The adaptive approach will simulate different subdivision levels then iteratively choose the best operation based on a geometric error cost function. Finally, the scalar micro displacement values are calculated by casting a ray along the interpolated displacement vectors into the original input mesh. The volume of the entire object is then optimized. Full details can be found in the original paper [8].

2 Earlier work

Bump maps [1] alter the surface normals of an object to make it look more realistic. A coarse mesh is used together with a bump map that contains height information that is added to the object surface during shading. It creates the illusion of a more complex surface while keeping the simple geometry unchanged.

In contrast, there are displacement maps [3, 2]. Similar to Bump Maps, an additional texture with height information is applied to offset an objects surface. Displacement Maps are actually changing the geometry of an object and not only the visual impression of it. There are various similar approaches to Bump and Displacement Maps that are adding additional details to 3D objects while leaving the original mesh unchanged [10].

To use the same model in different levels of detail, a progressive representation of the model can be used. One approach is using a progressive mesh where the

amount of vertices is reduced by edge collapsing [6]. Each collapse operation is encoded to be able to transition between different level of detail without losing information.

Normal meshes [5] have a similar idea as NVIDIA's micro meshes. A multi-resolution scheme is used where each level has a more dense mesh. The vertices in one level are displaced along the normal direction of vertices in the previous level. Thus, only one displacement scalar for every vertex has to be stored. On the other hand, this approach is not well suited for hardware acceleration because the previous level of the multi-resolution scheme is needed to calculate the next one.

One approach that is better suited for GPU accelerated computation is the dynamic mesh refinement [7]. Here the mesh is refined on the GPU by predefined patterns. During this process, new vertices are added to the mesh and new triangles are defined. For ray tracing, these triangles have to be added into an existing acceleration structure. This is usually too costly to be done in real-time raytracing applications. Thus, this approach is not well suited for ray-tracing applications.

The new mesh representation by NVIDIA[1] is similar to a displacement map approach. A highly detailed 3D mesh is compressed to reduce its footprint in GPU memory while preserving its complex surface. In contrast to previous techniques, micro meshes are supposed to perform well with real-time ray tracing. To get a micro mesh object, the original mesh has to be converted into the new format. In addition to the tools provided by NVIDIA[1], a different conversion algorithm has been proposed [8]. While there has been a comparison between the MMT pipeline and previous compression techniques that are not tailored to micro meshes [8], there has been no comparison between different micro mesh compression tools themselves. In the following, both conversion algorithms are compared in terms of various criteria.

3 Methods

Both, the NVIDIA and the MMT toolkit, differ in various aspects. We are comparing them using the following criteria: functionality, output quality, compression rate, performance, and automation. We expect that eventually the conversion from traditional 3D models into micro meshes will be integrated into existing artist tools (as already announced by NVIDIA[2]). We choose the comparison criteria to provide decision guidelines for two use cases: 1. generating micro meshes with the current toolkits, and 2. implementing a compression pipeline into other software.

Both compressing pipelines are only available as source code[3,4]. For the comparison, we build and install the tools on a Ubuntu 22.04 system using an NVIDIA Tesla T4 GPU, 16 GB RAM, and Intel(R) Xeon(R) CPU @ 2.00GHz 4

[2] `https://www.nvidia.com/en-us/on-demand/session/gtcspring23-s51410/`
[3] `https://github.com/NVIDIAGameWorks/Displacement-MicroMap-Toolkit`
[4] `https://github.com/NVlabs/micromesh-tools`

Cores (each 2 Threads). While the micro mesh tools offer support for processing textures and materials, we will only focus on the geometric aspects of micro meshes.

When compressing a 3D mesh with the given tools, there are a variety of different settings. For the experiments regarding compression and performance, we leave as many setting to its default value as possible. The reason for that is 1. comparing all possible setting combination would be beyond the scope of this work, and 2. we are interested in usability of the given tools for artists and for automated solutions. Thus, many settings could make the automated processing of 3D meshes more difficult.

One notable setting, we do adjust, is the number of triangles of the base mesh. While the MMT pipeline provides an automated base mesh generator, the NVIDIA tool needs a user defined threshold. We choose the NVIDIA base triangle count to match the MMT toolchain. If possible, we reduce the triangle count of the NVIDIA tool as long as there are no significant artifacts visible in the output mesh.

For comparing the compression rates, all files that are generated by the tools are combined to calculate the total file size. Usually the tools are generating three files. One including the base mesh and two other files including the displacement information.

3.1 Models

To compare both pipelines we are using a set of 3D models with different properties. In total six models are being used (see Table 1). Five of them are taken from online sources for reference models used in computing science. In addition, we generated one model. This generated model is based on a uniform square grid where every node of that grid gets assigned a random value in the range $[0, 1]$.

Fig. 2. Screenshot of some of the models used for comparison. From left to right: Bunny, Wall, Teapot, Noise, Ogre.

To ensure that the file size of all models are comparable, we converted them to `.obj` files using the free software Blender[5] before using them with the micro mesh tools. The export settings for all models were the same. In addition, we removed all data with information about texture, materials etc that are not related to the geometric properties of the object.

[5] `https://www.blender.org`

Table 1. Overview of the models being used for comparison.

Model	Source	Triangles	File size
teapot	Utah Teapot[6]	6,320	540.6kB
ogre	Keenan Crane[7]	39,856	1,331.2kB
noise	custom generated[8]	45,000	1,228.9kB
bunny	Stanford University Computer Graphics Laboratory[9]	69,451	6,734.4kB
wall	NVIDIA Developer Platform[10]	152,360	10,649.6kB

4 Results

4.1 Functions and Usability

Both pipelines offer the same basic functionality to compress input meshes into micro meshes. They can also be used to visualize the micro meshes and to adjust the compression settings. Both tools have additional visualization tools to analyze the micro mesh and to inspect the chosen subdivision levels and displacement vectors.

One important difference are the export options. Both tools can export the generated micro mesh to a new file format introduced by NVIDIA. The current issue with that file format is that is not widely supported by other 3D modeling software yet. An official support for the `glTF` file format is currently pending[11]. The MMT tools can export the micro mesh into different file formats including one that takes the micro mesh and turns it back into a traditional 3D model format. That model can of course not take any advantage of the benefits of micro meshes, because it is not longer in that compressed format but it shows how the micro mesh would look like when being rendered. This can particular be of interest when comparing the original model with the micro mesh. It is very difficult to deeply analyze a micro mesh model without the support of more advanced 3D modeling software.

4.2 Quality

Based on our experiments, the MMT pipeline seems to offer much better and more consistent quality. Every model that was compressed by the MMT tools can replicate the high details of the input mesh very well. We were not able to find any significant differences between the original and the compressed model (e.g. see Figure 3).

[6] `https://users.cs.utah.edu/~dejohnso/models/teapot.html`
[7] `https://www.cs.cmu.edu/~kmcrane/Projects/ModelRepository/`
[8] `http://graphics.stanford.edu/data/3Dscanrep/`
[9] `https://github.com/ettoh/sc23`
[10] `https://github.com/NVIDIAGameWorks/Displacement-MicroMap-Toolkit`
[11] `https://github.com/KhronosGroup/glTF/pull/2273`

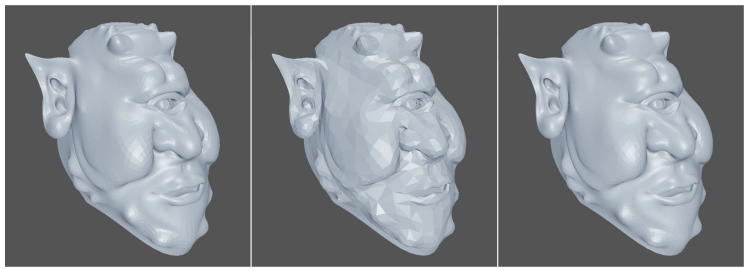


Fig. 3. Example results for the official MMT pipeline. Original mesh (left), base mesh (middle), and base mesh with displacement vectors applied (right).

One noticeable difference between original and compressed model are softer edges in the compressed model (see Figure 4). In all compressed models we could find that some edges of the compressed model are slightly softer than in the original model.

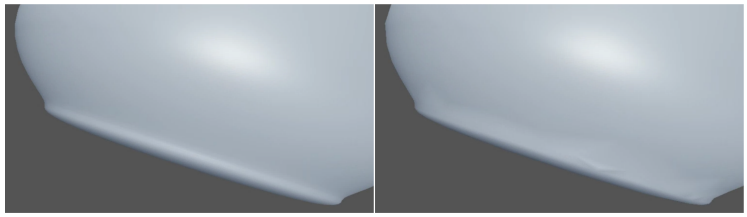


Fig. 4. Softer edges after compression with MMT pipeline. Left: original, right: micro mesh.

While the MMT pipeline offers consistent quality, the NVIDIA pipeline was producing both good and bad quality outputs during the experiments (see Figures 5 and 6). The quality of the compressed models seems to depend on the original input model, the number of triangles for the base mesh, and luck. The latter is caused by the nondeterministic behavior of the NVIDIA pipeline. Compressing the same input model with the same settings can lead to different outputs. Running the compression again can lead to a better or a worse model. During the experiments, both cases appeared. In addition, some models, like the `wall` model, seem to be better suited for the NVIDIA pipeline than others (e.g. the `teapot` model). The `wall` model provided similar consistent quality as the MMT pipeline while the other models showed significant artifacts in the output (e.g. Figure 5).

These artifacts seem to be related to a suboptimal base mesh. Therefore the number of triangles used for the base mesh can have significant impact on the quality (see Figure 6). With a more detailed base mesh, a higher quality micro mesh can be generated while reducing the compression rate. When choosing the number of base triangles similar to the MMT pipeline compression, the NVIDIA pipeline always yields worse quality than MMT (except the `wall` model).

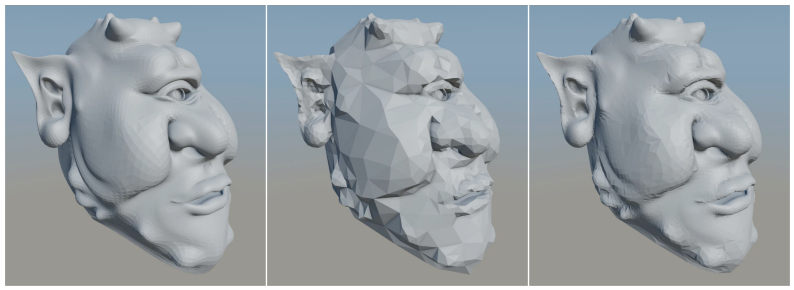


Fig. 5. Example results for the official NVDIDA toolkit. Original mesh (left), base mesh (middle), and base mesh with displacement vectors applied (right). Same base triangle count as the MMT pipeline (see Figure 3).

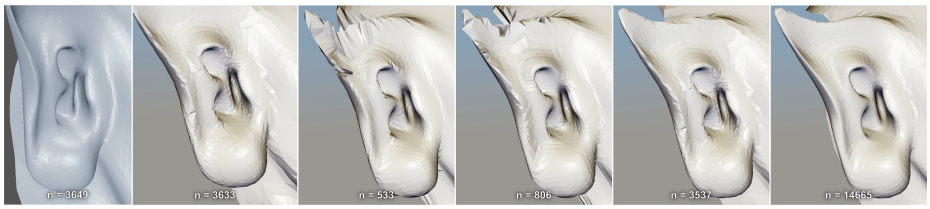


Fig. 6. Closer look to the compressed Ogre model. From left to right: MMT pipeline with automatically chosen base triangle count, NV pipeline with roughly the same triangle count, four NV compressed models with varying base triangle count (533, 806, 3537, 14665).

4.3 Compression

Both pipelines seem to yield better compression rates with increasing complexity (triangle count) of the input (see Table 2). Especially the NVIDIA pipeline seems to perform much better with higher resolution inputs than low resolution inputs. While difference between MMT and NVIDIA is getting smaller with increasing input complexity, the MMT pipeline generally provides better compression rates than the NVIDIA pipeline in our experiments. Notably, the NV pipeline seems to perform better for the Bunny model in terms of triangle count. This is because the triangle count in the NV pipeline has to be chosen manually. For the MMT pipeline it is chosen automatically. The bunny is the only model that provided an artifact-free compression with a lower base triangle count than the MMT pipeline. This could be because of the more uniform shape of the bunny model compared to the other models tested.

4.4 Performance

The NVIDIA pipeline is designed for running on a GPU. It takes advantage of the great parallelism of a GPU. It is specifically engineered for performance. Based on our experiments we can confirm that the NVIDIA pipeline offers great performance running on a GPU (see Table 3). The MMT pipeline runs on a CPU.

Table 2. Compression rates for the NVIDIA (NV) pipeline and the MMT pipeline with respect to file size and base mesh triangle count. The values in brackets are the ratios $\frac{compressed}{original}$ for file size and triangle count.

Model	Size NV	Size MMT	Triangles NV	Triangles MMT
teapot	345.5kB (63.91%)	127.2kB (23.53%)	2,378 (37.63%)	2,105 (33.3%)
ogre	452.5kB (33.99%)	270.8kB (20.34%)	3633 (9.12%)	3649 (9.16%)
bunny	439kB (6.52%)	375.1kB (5.57%)	1,310 (1.89%)	4,642 (6.68%)
wall	809.3kB (7.6%)	573.2kB (5.39%)	6,090 (4.0%)	5,627 (3.69%)

While the MMT pipeline also uses parallelism on the CPU, a GPU will usually outperform a CPU when it comes to parallelism. Therefore, it is very difficult to compare the performance of both pipelines because they are not executed on the same hardware. In the experiments, the MMT pipeline used significantly more time than the NVIDIA pipeline. Compressing the `wall` model with 152,360 triangles took almost 20 minutes, while the NVIDIA pipeline finished in less than 2 seconds. Based on that difference in performance and considering a commodity hardware setup using a dedicated GPU, we assume that the NVIDIA pipeline would outperform the MMT pipeline.

Table 3. Runtime for compressing each model with the NVIDIA (NV) pipeline and the MMT pipeline.

Model	Runtime NV	Runtime MMT
teapot	2s	158s
ogre	1s	450s
noise	$<$ 1s	386s
bunny	2s	558s
wall	2s	1143s

4.5 Automation

Due to the fact that micro meshes were just recently introduced with the NVIDIA GeForce RTX® 40 Series GPUs, currently 3D models are not created with the new format in mind. Therefore they have to be converted first. Making sure that the compressed 3D objects are fulfilling quality requirements could be a time consuming task depending on the amount of models to compress. Therefore it would be beneficial if the tool used for compression minimizes the manual adjustments needed to yield a satisfying micro mesh.

As seen in Figure 6, the number of base triangles can have a significant impact on the output quality. The MMT pipeline uses cost functions to determine the number of base triangles. No manual user input necessary. The NVIDIA pipeline relies on a user input. In the experiments the MMT tools provided great quality outputs using the default settings with no manual adjustments. The NVIDIA

tools provided mixed quality outputs and depends on manual settings like the base triangle count.

5 Conclusion

In conclusion, we have introduced the micro mesh format for 3D models that can compress high resolution models while preserving their details and being well suited for raytracing applications. We presented two currently existing pipeline tools that can convert conventional 3D models into the new compressed format. The NVIDIA[1] pipeline and the MMT pipeline [8].

We compared both pipelines regarding their usability, compression rate, quality, performance, and automation. We found that the MMT pipeline can provide consistent high-quality outputs with no manual adjustments necessary. The NVIDIA tools on the other hand can require more parameters to be tweaked to get a better result. It also uses nondeterministic algorithms that can make it more difficult to get consistent results. Performance-wise, both pipelines are difficult to compare. We assume that the NVIDIA pipeline offers a much higher performance due to the usage of GPU acceleration.

Based on our findings, we suggest the NVIDIA toolkit being used in cases where a human (e.g. an 3D artist) is compressing a 3D model. In these cases an immediate result (given by the high performance of the NVIDIA pipeline) can be crucial. The artist can therefore adjust the generated micro mesh and tweak parameters in accordance to the output. Especially when compressing huge models, the NVIDIA tools can provide enough performance to compute a result that can be tested immediately.

The MMT pipeline provides better and more consistent quality, and slightly better compression rates. Therefore we suggest it being used in an offline environment where a lot of 3D models have to be compressed without the need of immediate results. The consistent quality with no manual parameter adjustments necessary could make it a useful tool to compress 3D models before deploying a product. In that way, an artist could work with the NVIDIA toolkit to get immediate results and after the model is finished, it is compressed with the high quality MMT pipeline afterwards.

5.1 Limitations and Future Work

Due to the novelty of micro meshes and the current lack of support for its file formats, comparing both tools is a difficult task. In future works, it could be interesting to compare the outputs of both tools with a numerical metric, like the geometric error. Further investigation on the impact of the compression settings and the input model are of interest. In addition to that, the performance could be evaluated on a greater variety of hardware.

References

[1] James F. Blinn. Simulation of wrinkled surfaces. *ACM SIGGRAPH computer graphics*, 12(3):286–292, 1978.

[2] Gordon Collins and Adrian Hilton. Mesh decimation for displacement mapping. In *Eurographics (Short Presentations)*, 2002.

[3] Robert L Cook. Shade trees. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 223–231, 1984.

[4] Michael Garland and Paul S Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, 1997.

[5] Igor Guskov, Kiril Vidimče, Wim Sweldens, and Peter Schröder. Normal meshes. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 95–102, 2000.

[6] Hugues Hoppe. Progressive meshes. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 111–120. 2023.

[7] Haik Lorenz and Jürgen Döllner. Dynamic mesh refinement on GPU using geometry shaders. In *WSCG '2008: Full Papers: The 16-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision in co-operation with EUROGRAPHICS*, pages 97–104. University of West Bohemia, Plzen, Czech Republic, 2008.

[8] Andrea Maggiordomo, Henry Moreton, and Marco Tarini. Micro-mesh construction. volume 42, pages 1–18, New York, NY, USA, 2023. Association for Computing Machinery.

[9] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation.* MIT Press, 2023.

[10] László Szirmay-Kalos and Tamás Umenhoffer. Displacement mapping on the GPU-state of the art. In *Computer graphics forum*, volume 27, pages 1567–1592. Wiley Online Library, 2008.

A comparison of trust towards an AI and a human author in the context of social media news

Kai Schäfer

Department of Computing Science
Umeå University, Sweden
kai.schaefer.1@study.thws.de

Abstract. This paper presents a study in which participants (n=42) rated the trustworthiness of social media news content, that were labelled as either AI-generated or human-generated, depending on the group the participants were randomly assigned to. The results did not prove that framing content as AI-generated or human-generated had an statistically significant impact on the trustworthiness of the content. However, it was noticeable, that there is an influence of trust in AI technology and trust in the respective social media platform on the perceived trustworthiness of the news posts regardless of who was labelled as the author. Beyond the quantitative results of this study, the need for further qualitative studies investigating the underlying factors for the perception of trustworthiness in such a scenario is emphasised.

1 Introduction

In today's digital age, social media is often used to access news, but the credibility of information remains a major concern [17]. Especially among younger users, who rely on social media for news, there is a simultaneous sense of caution about the reliability of the information found on these platforms [20], [15]. The increased number of fake news challenges this concern even more, making it difficult to differentiate fact from fiction [8].

While a high amount of fake news were spread through social bots [4] computer programs designed to mimic human behaviour for opinion manipulation [2] media companies are now integrating artificial intelligence (AI) for autonomous news content creation [12], [7]. AI's role in journalism has evolved from handling simple news to more complex articles [1], [3].

Given the influential role of media companies in shaping public opinion and their reliance on perceived credibility [3], it is crucial to understand how differently people perceive a human author in comparison to an AI author in the context of social media news. This study addresses this gap by investigating public perceptions of the trustworthiness of news posts on the social media platform X, formerly known as Twitter, when explicitly informed that the news items were created either by an AI or a human journalist. Through a quantitative survey with an experimental design, participants were divided into two groups one informed of AI authorship, the other of human authorship and queried about

Suna Bensch (Eds.): Umeå's 27th Student Conference in Computing Science USCCS 2024, pp. 99–110, January 2024.

the trustworthiness of the news content. The study aims to clarify the influencing factors of credibility assessment, examining how transparent communication about the origin of news articles influences the perceived trustworthiness on social media posts.

2 Background

2.1 Automated journalism

Automated journalism describes the creation of journalistic content using algorithms [10]. Human language is created from structured data and published as a journalistic article. Automated journalism is currently mainly used to create short news articles that only contain text. Topics with a lot of data, such as financial reports, traffic updates and simple election result reports, are particularly suitable [10]. Within this change, the function of human journalists is evolving [16]. They are transitioning from being direct content creators to indirect producers, whereby they no longer personally generate content but instead provide algorithms with precise instructions, enabling these algorithms to produce journalistic material [16]. Currently, automatically generated news items are still characterized by repetitive language, which is why they tend to be used for smaller audiences [10]. Automated journalism still lags behind the current possibilities of human language creation. However, it is expected that this area will be significantly further optimised and used in other areas in the coming years [10].

2.2 Importance of X for journalism

Even during the time it was known as Twitter, the social media platform X was used not primarily for building social connections but for the consumption and distribution of information, especially news [18]. The dense network between Twitter accounts promotes the rapid dissemination of information, whereas the news and entertainment value of information is defined on X in a similar way to traditional mass media. Additionally, the majority of news companies use X as a secondary use of their content, which makes X an alternative medium to other news sources for the people. The micro-blogging platform is often the first contact point for news, especially for people with an affinity for technology and an interest in topics related to the internet [18]. However, besides valid news, the social media platform also spreads a high number of baseless rumors [14]. Although one study concluded that people do indeed question false news most of the time, it cannot be guaranteed that everyone will identify the false information [14]. Given the influential role of X as a central platform for consuming news and the spread of false information, it is important to investigate how people perceive the trustworthiness of news posted on the platform.

3 Literature review

In the context of understanding a persons trustworthiness towards online information one notable study introduced an integrated layer model of trust [13]. The model states that trust in information is linked to trust in its source. According to this model, trust in the source is influenced by trust in the medium, and this relationship is shaped by a more general propensity to trust. This should be considered while evaluating the credibility of information taking to account how trust is formed by different layers [13].

This layer model of trust was also used in a separate paper [11], that aimed to examine the impact of transparent communication about AI-generated Instagram and news websites content on trust. The trustworthiness of the source was recognized as a crucial element that strongly impacts trust. Additionally, users' familiarity with social media platforms and their experience with AI technologies were acknowledged as contributing factors. The study suggests that explanations about AI content creation may have limited influence on trust, while the significance of the content itself, whether generated by AI or human, is often perceived to be more influential than the authorship [11].

Furthermore contextual elements, including familiarity with AI and the frequency of social media usage, were identified to have an influence on trust. The frequency of engagement with social media demonstrated a considerable effect on evaluations of credibility, underscoring the importance of incorporating such contextual factors in trust-related research. Additionally, the study noted that articles in the topic of politics and economics generally received higher ratings than those in the entertainment field, irrespective of the medium and source [11].

Besides social media, comparisons of AI-generated and human made content were also made in the field of education. A study investigated the differences of educational content made by an AI and the one made by students. The results point out that AI-generated resources are perceived to be of equivalent quality to those generated by students, suggesting that AI-generated resources can serve as viable supplementary material in certain contexts [6].

Another study [9] provided helpful insights what people expect from human made news content in comparison to AI-generated content and how these expectations were met. Within the experiment participants expected more from human-written news in terms of readability and quality; but not in terms of credibility. Although these expectations of quality were rarely met. When participants saw only one article, differences in the perception of automated and human-written articles were small. However, when presented with two articles at once, participants preferred human-written news for readability but automated news for credibility [9]. These findings validate the results from another study [21], which showed that people's first expectations for the quality of human-written news might be too high and for automated news too low. Therefore, when being confronted with the actual content, people would adjust their quality ratings downwards for human-written and upwards for automated news [21].

4 Method

To determine whether people place more trust in an AI author or a human author, an online survey was conducted. The platform LimeSurvey was used for this, as it provides the desired functionality and has good support for data protection. The answers were anonymous and stored on a local server. The participants also agreed to the privacy policy. A total of 42 people took part in the survey. 20 of these were male, 21 female and 1 non-binary. The majority of respondents were 19-35 years old and had a higher academic degree. The participants were mostly acquired from the author's immediate environment as well as through survey platforms.

The survey consisted of two parts. The first part included questions about the participant's demographic characteristics and their propensity to trust. Participants were also asked about their trust in artificial intelligence and the social media platform X. In the second part, an experiment was conducted in which the participants were divided into two groups. Both groups were shown a screenshot of a news feed of an imaginary account on X. One group was told that the account was run by journalists. The other group was told that this was a profile that posts AI-generated news. It was not specified exactly how the AI receives the data and what methods it uses to create the content, as a study found that increased transparency in the way AI-content is created does not have a significant impact on trust [11]. Figure 1 shows the presented news posts both to the AI and the human-journalist group. The topics were selected from various fields like science, politics and entertainment to reduce the variance in the results due to a biased perception of trustworthiness of a person towards one specific section of news. The selection should rather represent the overall news-feed from the author instead of focusing on one post.

The X account 'BCC News' of the British Broadcasting Corporation (BBC) was used as a template for the news items shown[1]. After the participants had viewed the screenshot, they were asked questions about the perceived trustworthiness of the news items and the author. The following questions were displayed to the group with the AI being the author.

- When you are looking for information, how often would you use the AI's X profile as opposed to other sources?
- What do you think is the credibility of the AI?
- How much do you trust the institute that made the AI?
- How much confidence do you have in the people who maintain the AI?
- If you are in need of information, how confident are you that you can find it on the profile of the AI?
- How large do you think is the risk of getting inaccurate information on the AI's profile?

[1] Profile of BBC News (UK) on X, available at https://twitter.com/BBCNews, last accessed on 06.11.2023

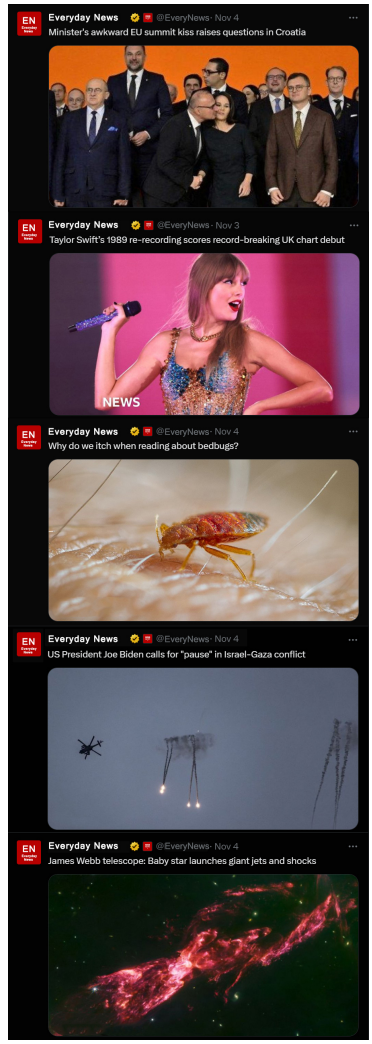


Fig. 1. Edited screenshots of a news feed on X used for the experiment

The results of the experiment were used to determine the difference between the authors. The questions asked in advance served to determine whether any differences in trust were actually due to the authors or whether there were other reasons. An important study shows that a person's trust in information depends on their propensity to trust, their trust in the medium (X) and their trust in the source (AI / human author) [13]. Based on this model, the influencing factors mentioned were surveyed. For the questions on the propensity to trust, the trust in the medium X and the trust in the author, the corresponding questionnaire of the study was used and only the name of the medium and the source were adapted [13]. Similar studies have also chosen such a structure [11].

In addition to the existing questionnaires, questions were added regarding the participant's demographic characteristics. Gender, age, work experience and educational level were asked, as studies have shown that these have an influence on the trustworthiness of a person [19, **?**]. According to a similar study the experience with AI and social media as well as the overall trust in news and news agencies also have a significant influence [11]. The participants were therefore asked to provide insights on these topics as well.

Subsequently, the survey results were analyzed specifically examining whether a significant difference existed in the trust levels attributed to the different authors. With a few exceptions, the 5-point Likert scale was used. Comparative values could be established for these questions and correlations between the results of the experiment and possible influencing factors could be investigated. For the open questions, where participants answered in text form, cluster methods were used. Notably, this study focused solely on the assessment of trust disparities and did not delve deeper into the underlying reasons for any observed differences, leaving such inquiries for potential future qualitative investigations.

5 Results

The data from the experiment for the two groups were compared with each other. Possible correlations between the results of the experiment and the influencing factors surveyed were also examined.

Table 1 shows how the participants perceived the trustworthiness of the two authors' social media news feed. The values regarding the question of the risk of getting false information were reversed, so that a higher value represents a lower risk. This was made to guarantee that a higher overall score represents a higher trustworthiness in the author. The comparison of the two groups showed no statistically significant difference between the trustworthiness in an AI author and the trustworthiness in a human journalist at a significance level of 0.05. The t-test showed a p-value of 0.45, indicating that the null hypothesis of equality of the means is not rejected.

The values of the individual questions in the experiment already provide an indication of how the participants perceived the trustworthiness. In order to identify further influencing factors, the results of the experiment were examined for possible correlations with the additional questions. The correlation coefficient ranges from +1 to -1, with a positive value representing a positive correlation and a negative value a negative correlation. The interpretation of the values follows Cohen's guidelines [5], which classify correlation values as weak (± 0.1 - ± 0.3), moderate (± 0.3 - ± 0.5), and strong (± 0.5 +). Table 2 shows that there is a strong correlation between the overall score for the trust in AI and the experiment results for the AI author. The correlation between the experience with AI and the experiment results for the AI author is classified as moderate as well as the correlation between the score for the trust in X and the experiment results. The remaining variables seem to have a rather low impact on the experiment results.

The answers for gender and educational level had to be transferred to a numeric value to calculate the correlation coefficient.

To get even more information out of the data, the values of the influencing factors were also examined for correlation. The results in Table 3 show that the gender has a weak influence on a persons propensity to trust and the trust in AI. The age has also a weak influence on a persons trust in AI. The educational level has weak influence on the propensity to trust as well as the social media use and the trust in X.

In addition to the correlations, the results of the different question categories in the survey are also relevant. The average overall value for the propensity to trust was 3,57, the average value for trust in AI was 2,97 and the average value for trust in X was 2,10. All the questions were answered with a value on the 5 point Likert scale making 1 the lowest and 5 the highest possible value.

Question	**AI (mean)**	**Human (mean)**
When looking for information, how often would you use the «author»'s X profile as opposed to other sources?	1,71	1,85
What do you think is the credibility of the «author»?	2,38	2,90
How much do you trust the institute of the «author»?	2,67	2,85
How much confidence do you have in the people who maintain the AI / that post the news?	2,91	2,75
If you are in need of information, how confident are you that you can find it on the profile of the «author»?	2,67	2,85
How large do you think is the risk of getting inaccurate information on the «author» profile? R	2,14	2,90
	2,41	**2,68**

Table 1. Mean of participant responses for their trust in the news items of both authors on a 5-point Likert scale. A mean of 1 would represent the lowest possible amount of trust, whereas a mean of 5 would indicate, that every participant chose the highest possible value for trust. Values below 3 therefore indicate a rather low trustworthiness.

Variable 1	**Variable 2**	**r-value**
Age	Experiment result	-0,05
Educational level	Experiment results	-0,04
Work experience	Experiment results	-0,08
Gender	Experiment results	-0,16
Score for 'Propensity to trust'	Experiment results	-0,03
Score for 'Trust in X'	Experiment results	0,33
Score for 'Trust in AI'	Experiment results (only AI)	0,55
Experience with AI	Experiment results (only AI)	0,42
Social Media Use	Experiment results	0,12
General trust in news	Experiment results	0,06
General trust in news companies	Experiment results	0,01

Table 2. Pearson's correlation coefficient between different variables and the experiment results. The correlation value r represents the amount of influence a variable has on the results of the experiment. The coefficient ranges from -1 to +1, with values above 0 indicating a positive correlation and values below 0 indicating a negative correlation. According to Cohen's guidelines [5], values between ±0.1 and ±0.3 are classified as weak correlations, values between ±0.3 and ±0.5 are classified as moderate correlations, and values of ±0.5 or higher/lower as strong correlations.

Variable 1	**Variable 2**	**r-value**
Gender	Score for trust in AI	-0,29
Age	Score for trust in AI	0,27
Gender	Score for Propensity to trust	-0,26
Educational level	Score for Propensity to trust	-0,29
Social Media use	Score for Trust in X	-0,18

Table 3. Pearson's correlation coefficient between different variables that could have an influence on the experiment results. The table only includes correlations that are above/below ±0.1 to limit the results to only include weak (±0.1 - ±0.3), moderate (±0.3 - ±0.5), and strong (±0.5 +) correlations according to Cohen's guidelines [5]. The r-value ranges from -1 to +1, with values above 0 indicating a positive correlation and values below 0 indicating a negative correlation.

6 Discussion

The results of the study do not provide statistically significant proof that a persons trust in social media news content is affected by framing it as either

AI-generated or human-generated. However, it is noticeable in Table 1 that the overall scores as well as the means for every question of the experiment are in both groups below 3 which indicates a rather low trustworthiness in the news items of both authors. The reasons for this can be very different. One influencing factor could be the overall trust in X, which was rated as rather low by the participants. The low level of trust in the news posts could therefore be attributed to the fact that the participants have little trust in the platform itself. The correlation coefficient between both values reinforces the assumption that trust in X has an influence on the trustworthiness of the news posts.

Following the layered model of the previously mentioned study [13], the trust in the social media posts depends not only on trust in X and the respective author, but also on the propensity to trust. This study, however, could not confirm this influence. The demographic values gender, age, work experience and educational level also appear to have no significant or only a very weak influence on the trustworthiness of the authors' news posts. While a moderate influence of age and gender on trust in AI, as well as the propensity to trust, was identified, it appears that this has only a minimal direct effect on how trustworthy the news posts were perceived. The low values of the experiment also do not appear to be a result of a generally low level of trust in news or news organisations. The study shows no statistical correlation in this respect. The participants' perception of most news and news organisations was neutral overall.

When analyzing the individual questions of the experiment, it is noticeable that the participants rated the credibility of a journalist slightly higher than that of an AI. They also rated the risk of receiving false information higher for an AI than for a journalist. These findings could be due to people's experience with AI and their general trust in AI. Less experience with AI indicates that the person may not understand exactly how this technology works. The person may therefore assume, that it is more likely to receive incorrect information, which also leads to a perception of lower credibility. This assumption is supported by the link between the score for trust in AI and the experiment results of the AI group. The results show a strong positive correlation between them. There is also a moderate correlation between the value of experience with AI and the experiment results of the AI group.

While this study employs a quantitative approach, it is important to note that there are qualitative studies needed to investigate the reasons for the observed low trustworthiness of the news items and the small difference between the authors. The inclusion of open-ended questions in the questionnaire revealed insights, that could function as a starting point for future qualitative research. Participants mentioned factors such as news post design, lack of company name recognition, and conditional trust in AI for non-political news as important. They also expressed concerns about the phrasing and topics of news, highlighting the multifaceted nature of factors influencing trust. The results of this study are therefore influenced by the selected news posts and the way they are presented. Consequently, this does not represent all human journalists or all social media profiles of an AI. Repeating the study method with a different representation of

the news posts could therefore bring more insights on the topic. The participants in the study could also be selected on the basis of other demographic factors. This study mainly represents people with higher academic qualifications between the ages 19 and 35. People of a higher age or a different cultural background may have different views on the topic.

7 Conclusion

This study has delved into the topic of trust in news content on social media, by investigating the impact of labeling content as either AI generated or human generated on the perceived trustworthiness. This was complemented by examining possible factors that shape the perception of trustworthiness towards the presented social media news items.

The experiment results revealed no statistically significant difference in the trustworthiness of content that was framed to be created by an AI in comparison to content that was framed to be human-made. The presented news items in the experiment where also perceived as rather low for both authors. Analyzing correlations between various factors and the experiment results unveiled some interesting findings. It was observed that the perception of news posts, that are labeled to be made by an AI, could be influenced by the overall trust in the AI technology, which could be formed by a person's experience and knowledge about AI. Additionally, the study identified the trust in the respective social media platform to have a possible influence on the perceived trustworthiness of the news posts independent of the author. Demographic values as well as a overall propensity to trust seem to have a rather low impact on the perceived trustworthiness. The participants named mainly design factors of the news post as influencing factors of their trust. The findings of the study therefore seem to align with the statement of a previous study saying that the actual content of a news post may be rather more important than the authorship [11].

While the results of the quantitative study provide valuable insights, it is crucial to acknowledge its limitations. Future qualitative research is needed to delve deeper into the underlying reasons for these observed trust perceptions. Moreover, the study's demographic focused on people with higher academic qualifications and a specific age range, which calls for a broader sampling to ensure a more representative understanding of the difference in the trustworthiness of an AI author compared to a human journalist.

In conclusion, this study contributes to the ongoing discourse on trust in automated news especially in the field of social media news. By shedding light on the perceived trustworthiness of people towards news posts the study emphasizes the need for continued exploration into the factors shaping public trust in digital journalism.

References

1. Elizabeth Blankespoor, Ed deHaan, and Christina Zhu. Capital market effects of media synthesis and dissemination: evidence from robo-journalism. *Review of*

Accounting Studies, 23:1–36, 2018.
2. Florian Brachten, Milad Mirbabaie, Stefan Stieglitz, Olivia Berger, Sarah Bludau, and Kristina Schrickel. Threat or Opportunity? - Examining Social Bots in Social Media Crisis Communication. In *Australasian Conference on Information Systems*, pages 406–412, 2018.
3. J Scott Brennen, Philip N Howard, and Rasmus K Nielsen. What to expect when you're expecting robots: Futures, expectations, and pseudo-artificial general intelligence in UK news. *Journalism*, 23(1):22–38, 2022.
4. Giovanni Luca Ciampaglia, Alexios Mantzarlis, Gregory Maus, and Filippo Menczer. Research Challenges of Digital Misinformation: Toward a Trustworthy Web. *AI Magazine*, 39(1):65–74, 2018.
5. Jacob Cohen. *Statistical Power Analysis for the behavioral sciences*. Academic Press, New York City, New York, 1977.
6. Paul Denny, Hassan Khosravi, Arto Hellas, Juho Leinonen, and Sami Sarsa. Can we trust AI-generated educational content? Comparative analysis of human and AI-generated learning resources. pages 1–12. arXiv (Cornell University), 2023.
7. Yair Galily. Artificial intelligence and sports journalism: Is it a sweeping change? *Technology in Society*, 54:47–51, 2018.
8. Petra Grimm, Tobias Keber, and Oliver Zöllner. *Digitale Ethik. Leben in vernetzten Welten*. Reclam, Bonn, 2019.
9. Mario Haim and Andreas Graefe. Automated News. *Digital Journalism*, 5(8):1044–1059, 2017.
10. Mario Haim and Andreas Graefe. *Journalismus im Internet: Profession - Partizipation - Technisierung*, chapter Automatisierter Journalismus. Anwendungsbereiche, Formen und Qualität, pages 139–160. Springer Fachmedien Wiesbaden, Wiesbaden, 2018.
11. Lennart Hofeditz, Milad Mirbabaie, Jasmin Holstein, and Stefan Stieglitz. Do You Trust an AI-journalist? A Credibility Analysis of News Content with AI-Authorship. In *ECIS 2021 Research Papers*, volume 50, 2021.
12. Bronwyn Jones and Rhianne Jones. Public Service Chatbots: Automating Conversation with BBC News. *Digital Journalism*, 7:1–22, 07 2019.
13. Teun Lucassen and Jan Maarten Schraagen. Propensity to trust and the influence of source and medium cues in credibility evaluation. *Journal of information science*, 38(6):564–575, 2012.
14. Marcelo Mendoza, Barbara Poblete, and Carlos Castillo. Twitter under Crisis: Can We Trust What We RT? In *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10, pages 71–79, New York, NY, USA, 2010. Association for Computing Machinery.
15. Milad Mirbabaie, Deborah Bunker, Stefan Stieglitz, Julian Marx, and Christian Ehnis. Social media in times of crisis: Learning from Hurricane Harvey for the coronavirus disease 2019 pandemic response. *Journal of Information Technology*, 35(3):195–213, 2020.
16. Philip Napoli. On Automation in Media Industries: Integrating Algorithmic Media Production into Media Industries Scholarship. *Media Industries Journal*, 1(1):33–38, 2014.
17. Christoph Neuberger and Christian Nuernbergk. Journalismus im Internet: Zwischen Profession, Partizipation und Technik. *Media Perspektiven*, 4:174–188, 2009.
18. Christoph Neuberger, Christian Nuernbergk, and Vom Hofe Hanna Jo. *Twitter und Journalismus: Der Einfluss des "Social web" auf die Nachrichten*. Landesanstalt für Medien Nordrhein-Westfalen, Düsseldorf, 2011.

19. Ayush B. Shrestha, Richard A. Bernardi, and Susan M. Bosco. The propensity to trust others: Gender and Country Differences. *Accounting and Finance Research*, 2(2):69–78, 2013.
20. Stefan Stieglitz, Milad Mirbabaie, Tobias Kroll, and Julian Marx. 'Silence' as a Strategy during a Corporate Crisis - The Case of Volkswagen's 'Dieselgate'. *Internet Research*, 29:921–939, 2019.
21. Hirini van der Kaa and Emiel Krahmer. Journalist versus news consumer: The perceived credibility of machine written news. In *Proceedings of the Computation+Journalism conference*, volume 24, pages 1–4, 2014.

Author Index