Socially Aware Path Planning for a Flying Robot in Close Proximity of Humans

HYUNG-JIN YOON, CHRISTOPHER WIDDOWSON, THIAGO MARINHO, RANXIAO FRANCES WANG, and NAIRA HOVAKIMYAN,
University of Illinois at Urbana-Champaign, USA

In this article, we present a preliminary motion planning framework for a cyber-physical system consisting of a human and a flying robot in vicinity. The motion planning of the flying robot takes into account the human's safety perception. We aim to determine a parametric model for the human's safety perception based on test data. We use virtual reality as a safe testing environment to collect safety perception data reflected on galvanic skin response (GSR) from the test subjects experiencing a flying robot in their vicinity. The GSR signal contains both meaningful information driven by the interaction with the robot and also disturbances from unknown factors. To address the issue, we use two parametric models to approximate the GSR data: (1) a function of the robot's position and velocity and (2) a random distribution. Intuitively, we need to choose the more likely model given the data. When GSR is statistically independent of the flying robot, then the random distribution should be selected instead of the function of the robot's position and velocity. We implement the intuitive idea under the framework of hidden Markov model (HMM) estimation. As a result, the proposed HMM-based model improves the likelihood compared to the Gaussian noise model, which does not make a distinction between relevant and irrelevant samples due to unknown factors. We also present a numerical optimal path planning method that considers the safety perception model while ensuring spatial separation from the obstacle despite the time discretization. Optimal paths generated using the proposed model result in a reasonably safe distance from the human. In contrast, the trajectories generated by the standard regression model with the Gaussian noise assumption, without consideration of unknown factors, have undesirable shapes.

CCS Concepts: • **Mathematics of computing → Kalman filters and hidden Markov models**; • **Computing methodologies → Motion path planning**; • **Computer systems organization → External interfaces for robotics**; *Robotic autonomy*; • **Human-centered computing** → *Virtual reality*;

Additional Key Words and Phrases: Human-robot interaction, hidden Markov model, optimal path planning

ACM Reference format:
Hyung-Jin Yoon, Christopher Widdowson, Thiago Marinho, Ranxiao Frances Wang, and Naira Hovakimyan. 2019. Socially Aware Path Planning for a Flying Robot in Close Proximity of Humans. *ACM Trans. Cyber-Phys. Syst.* 3, 4, Article 41 (September 2019), 24 pages.
https://doi.org/10.1145/3341570

This material is based upon work supported by the National Science Foundation under National Robotics Initiative Grants No. 1528036 and No. 1830639.

Authors' addresses: H.-J. Yoon, T. Marinho, and N. Hovakimyan, University of Illinois at Urbana-Champaign, Mechanical Science and Engineering, 1206 W. Green Street, Urbana, IL, 61801; emails: {hyoon33, marinho, nhovakim}@illinois.edu; C. Widdowson and R. F. Wang, University of Illinois at Urbana-Champaign, Psychology, 603 E. Daniel Street, Champaign, IL, 61820; emails: {widdwsn2, wang18}@illinois.edu.

2378-962X/2019/09-ART41 $15.00
https://doi.org/10.1145/3341570

1 INTRODUCTION

Multirotor copters have become popular as commercial, industrial, and educational platforms. The mechanical simplicity and agile maneuverability appeal to the applications in the urban environment, such as media production, inspection, and precision agriculture. Having these micro unmanned aerial vehicles (UAVs) in our everyday lives can be beneficial. For example, by deploying fast UAVs, e-commerce retailers like Amazon and Walmart can reduce logistics costs. In addition, by leveraging their agility and reliability, applications in elderly care and mobile surveillance are being developed. In all these examples, it is necessary to fly near people and navigate in densely populated areas. In contrast to the current mobile robots that autonomously operate without considering humans, these flying robots should be developed to interact and cooperate with people. Traditionally, robot control and motion planning have focused on the robot's actual safety, i.e., the ability to generate safe paths that avoid collisions with obstacles. However, it is also important for humans to feel safe when flying robots operate around them. Studies of human perception have shown that there is a sharp distinction between human perceived safety and actual safety. This article presents a path planning framework, which (i) approximates a safety perception model as a function of the robot's position and velocity and (ii) generates collision-free trajectories taking account of the safety perception model.

The system consisting of flying robots and humans in an urban environment is a cyber-physical system (CPS). However, the CPS inherently has hidden states due to human involvement. The incomplete observation of the state of the human makes it challenging *to learn* the safety perception model. The human's response to psychological experiments sometimes shows unexpected behavior, e.g., the change of the focus of attention can make the response independent of the test variable. Typically, we attribute possible causes of unexpected behaviors to unknown factors. Naively assuming the unknown factor to be an independent identically distributed (i.i.d.) Gaussian noise model would not be suitable for the data, where only partial observation of the state of the system is provided. However, the i.i.d. Gaussian noise assumption is popularly used for regression tasks, since maximizing the likelihood conveniently reduces to the mean-squared error (MSE) minimization problem. For example, a recurrent neural network (RNN) was trained to predict the music mood (valence) by minimizing mean-squared error (Weninger et al. 2014). The data fitting resulted in a low coefficient of determination at most 50%. The undesired *goodness of fit* despite the complex model with a lot of parameters in the RNN suggests that there are other factors not contained in the data that may influence the outcome. To address this issue in a general case, we propose to use a model where the focus of the attention is modeled as a hidden variable in a hidden Markov model (HMM). The HMM divides the data samples into two clusters: (i) relevant samples where the target variable (arousal) can be predicted as a function of the feature (robot's position/velocity); (ii) irrelevant samples where it is better predicted by a random source than a function of the feature.

The data-driven model of human's response to the flying robots can be used under the optimal control framework. However, the optimal trajectory generation in complex environments is a challenging problem. The problem is more complicated when the environment is inhabited by humans, whose safety perception needs to be accounted for as well. To name a few existing methods, Kinodynamic RRT* was proposed in Webb and van den Berg (2013) to deal with arbitrary cost functions. The pseudospectral (PS) method (Elnagar et al. 1995) discretizes trajectories into the polynomial functions with finite order, and then the optimal trajectory generation problem becomes a finite-dimensional optimization problem. The PS method has been applied successfully for solving trajectory optimization problems (Bollino and Lewis 2008). As a result of discretization, the PS method can verify the constraints, including collision avoidance, only at the discretized time nodes. As collision avoidance is a critical requirement, a very dense discretization mesh would be

needed. The computation of optimal trajectories can become intractable and expensive for implementation in CPSs. Using Bézier curve representation[1], convex hulls that contain the trajectory can be used for collision verification along the entire trajectory without increasing the number of nodes (Choe et al. 2013). In Cichella et al. (2018), it was shown that using Bernstein polynomials the optimal control solution can be approximated sufficiently closely as the number of time-nodes increases, while at the same time ensuring spatial separation from obstacles. In this article, we represent the trajectories by both Bézier curves and polynomials with Legendre-Gauss-Lobatto (LGL) time nodes. Such representation helps to avoid collisions while considering the human's safety perception model.

1.1 Related Results and Statement of Contributions

Human's perception of a flying robot dependent upon its spatial and temporal variables (e.g., distance and speed) has been studied using virtual reality testbed, and as well using a real flying robot. A comfortable approach distance for a flying robot was studied in (Duncan and Murphy 2013), where the authors tested the effects of the size of the robot on the comfort levels of the human subjects using behavioral, physiological and survey measures. Distancing with the robot and the interaction preference between two differently behaving robots were investigated for speed and repeating behavior (cyclicity) using VR experiments (Duncan and Murphy 2017). A comparison of comfort with a small UAV versus a ground vehicle was studied through tests measuring comfort distance (Acharya et al. 2017).

On the other hand, various design approaches to the *human-aerial-robot* system have been explored again to ensure comfort for humans. Laban effort[2] was employed to design flight paths for a flying robot in Pakrasi et al. (2018) and Sharma et al. (2013), and the effect on arousal and valence due to the design parameter of the flight path was tested. Emotional encoding in a flight path of a robot was investigated in Cauchard et al. (2016), where the encoding was derived from characterizing stereotypes of personality and motion parameters using *interaction vocabulary*. In Szafir et al. (2015), the authors proposed a flight path design approach, which improves the ease of human's perception of the robot's motion, and the proposed design is tested using survey measures. A signaling device that resembles the turn signals of a car was proposed to communicate the robot's intent to humans (Szafir et al. 2014). Using gestures to communicate the user's intent to the robot was investigated in Cauchard et al. (2015).

Socially aware navigation for ground robots has been more extensively studied in the last decade compared to UAVs (see Charalampous et al. (2017) and Kruse et al. (2013) for review). Although human-aware navigation is a broad concept, most papers focus on improving the robot's behavior in the aspects defined as follows (Kruse et al. 2013):

- Comfort is the absence of annoyance and stress for humans in interaction with robots.
- Naturalness is the similarity between robots and humans in low-level behavior patterns.
- Sociability is the adherence to explicit high-level cultural conventions.

This article focuses on improving the comfort of the user who interacts with flying robots. Human's perception of safety is different from objective safety. Even when a robot is designed to ensure safe operation, a human user may still feel that the robot's motion is not safe due to a lack of trust in the technology. Proxemics theory (Hall 1966) has been extensively accepted by roboticists

[1]Bézier curve is a curve determined by a polynomial trajectory, where the polynomial is written in the form of Bernstein polynomial (Lorentz 2012). The coefficients of the Bernstein polynomials are also called control points of the Bézier curve. The convex hull of the control points contains the Bézier curve.
[2]A method to interpret human motion used in choreography (Laban and Ullmann 1971).

and led to various ways of modeling human space in the context of socially aware robotics. In Vega et al. (2019), using the Gaussian kernel as a spatial model to consider personal space, the human-occupied space is probabilistically constrained following the proxemics theory. In Kostavelis et al. (2017), a Gaussian mixture model was used to consider the constraint space for groups of people where each Gaussian kernel presents a person's space. Furthermore, Papadakis et al. (2014) and Sisbot et al. (2007) used semantic labeling of the 3D maps or camera images following the human space modeling theory. In Sisbot et al. (2007), a human-aware motion planner regulates the distance between robots and humans by applying a social mapping that considers human's states such as sitting and standing. In Papadakis et al. (2014), the authors proposed an algorithm that addresses social mapping as density estimation. The estimated density indicates the constraint space for robot navigation.

Conforming spatial constraints that follow social conventions might not be enough to ensure the comfort of the humans. In Butler and Agah (2001), the experiment showed that the human's feeling of discomfort also depends on the speed of the robots. Also, there are cultural differences that need to be considered in social navigation (Joosse et al. 2014). Beyond the proxemics theory-based social navigation, there exist learning-based approaches (Luber et al. 2012; Vasquez et al. 2014) that consider the complex nature of the socially aware robot navigation. In Luber et al. (2012), using surveillance camera images on pedestrians, the authors estimate a pedestrian model to be used with a motion planner. In Vasquez et al. (2014), the authors proposed to use inverse-reinforcement learning to identify the social norm in human's navigation in terms of cost to minimize in optimal control. In Chen et al. (2017), reinforcement learning was applied for navigation control of a nonlinear cost function that is congruent to a social norm.

Contribution. In the papers cited above, the focus is on either discovering a general model in *human-aerial-robot* interaction based on the empirical data or devising a heuristic method to improve the acceptability of the robot for humans. Many social navigation methods for ground robots aim to avoid constraint space that is determined based on proxemics theory. Nevertheless, it is not clear whether avoiding intimate space around a human is ultimately the desired behavior of social robots. There are learning-based methods for social navigation that do not rely on the proxemics theory. However, the learning-based approaches in Luber et al. (2012) and Vasquez et al. (2014) focus on how to predict human's motion instead of predicting how humans would feel when the robot is operating around them.

The contributions of this article are listed as follows:

(1) We conduct experiments to study human's safety perception in the presence of a flying robot in proximity using the physiological sensor signal (Galvanic skin conductance) measured in virtual reality (VR) as a safe testing environment.
(2) We devise a hidden Markov model (HMM) approach to obtain a data-driven model of the safety perception model, which is a map from the robot's position and velocity to Galvanic skin response (GSR). The estimated function is more suitable for engineering optimization than the sparse hypothesis testing used in previous results of human-robot interaction (HRI).
(3) We propose to use Bernstein polynomials to discretize the trajectories of a robot. The Bernstein polynomials ensure collision avoidance between time nodes despite the discretization while optimizing the nonlinear cost function that considers the estimated safety perception model.

The remainder of the article is organized as follows. In Section 2, the VR experiment set-up is described, and the preliminary finding that relates the physiological arousal to the perceived safety

Fig. 1. Flying robot observed in the VR environment (an illustration video at https://youtu.be/XnaXzdHlxUA).

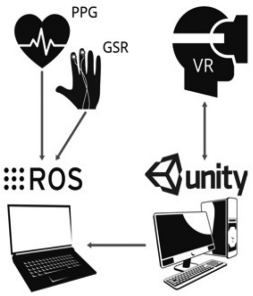


Fig. 2. Data acquisition using ROS (Quigley et al. 2009) and Unity (Team 2010).

is introduced. In Section 3, we propose a model to address the influence of unknown factors and validate it against the VR experimental data. In Section 4, the optimal path planning that takes the human arousal model into account is presented. Section 5 summarizes and discusses future directions.

2 VR EXPERIMENT AND DATASET GENERATION

Virtual Reality offers a safe, low-cost, and time-efficient method to collect data (Duncan and Murphy 2017). For example, the precise coordinates of the human and robot can easily be recorded in real-time, which is useful for studying spatial-temporal variables in human behavior. To this end, we developed a VR test environment to explore human-aerial vehicle interactions in a variety of experimental scenarios (Marinho et al. 2016; Widdowson et al. 2017). Concurrent psychophysiological reactions of participants are recorded in terms of head motion kinematics and electrodermal activity (EDA) and are time-aligned with attributes of the robot's flight path, e.g., velocity, altitude, and audio profile. We developed a data acquisition system, where the *robot operating system* (ROS) assigns time stamps to data packets from the sensors, as shown in Figure 2 (Quigley et al. 2009).

The virtual environment (VE) was developed with the Unity game engine (Unity 5.4.1f1) and presented on an HTC Vive VR headset connected to a Windows 10 desktop computer (i7-5820K, 3.3GHz CPU; 32GB RAM; NVIDIA GeForce GTX 980 Ti graphics card). The head-mounted display consists of two low-persistence AMOLED displays (90Hz) with a combined resolution of $2{,}160 \times 1{,}200$ ($1{,}080 \times 1{,}200$ per eye) and approximately 110° horizontal field of view. Fully spatialized audio was implemented for all sound sources including UAV flight and ambient noises.

Across three experiments, participants passively observed UAV trajectories in a simulated VR environment. UAV flight paths were manipulated in terms of velocity, altitude, and acoustic profile to examine their effect on physiological arousal and head motion kinematics. During the simulation, participants were seated at the junction of a three-way intersection in a simulated urban environment. Three trajectories and their mirror reversals were fit to the environment for a total

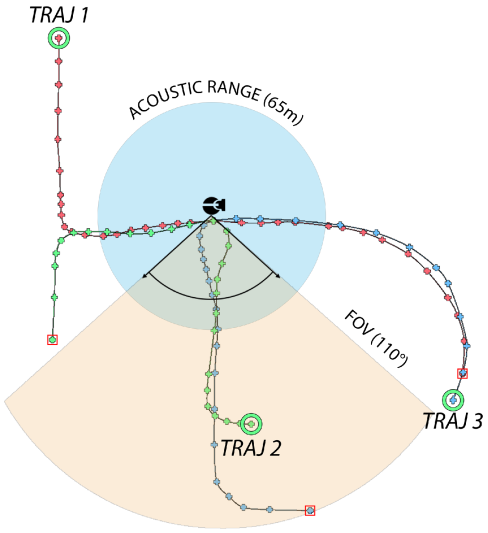


Fig. 3. Designed UAV trajectories.

of six unique trajectories; average path length, 208 $[m]$; altitude, 1.6 $[m]$ as shown in Figure 3. The simulation started with a 90s baseline period without any UAV flight, allowing time for the GSR signal to plateau. The first UAV then appeared and completed its trajectory past the subject. After a brief pause, the next UAV flew past, and so on for all six trajectories. The entire experiment lasted approximately 30min. Figure 4 shows the position and velocity profiles of all these flying robots. We collected the data from 56 participants (20 males, 36 females) recruited from our university. A detailed report of the virtual environment (VE) system used for human subject testing with small UAVs can be found in Widdowson et al. (2017).

The skin conductance signal is preprocessed by EDA analysis package, *Ledalab*, to generate the phasic activation signal (Benedek and Kaernbach 2010). The EDA toolbox decomposes the skin conductance signal into the phasic and tonic signal, as shown in Figure 5. The phasic signal is then deconvolved to determine phasic activation; phasic response represents changes in sympathetic arousal due to event-related activation.

To the best of our knowledge, there has been no standard index of perceived safety in the literature. Although physiological measurements of arousal (e.g., EDA) alone are not necessarily equivalent to people's perceived safety, several pieces of evidence suggest that the EDA measure of physiological arousal in our study is closely related to people's anxiety induced by the approaching drone. For example, in a follow-up experiment examining the effects of path height on people's EDA responses, we found that the EDA phasic response was significantly stronger for a drone approaching at eye-height where a potential collision was possible than when the drone was flying at a height beyond the observer's head where there is no danger of collision. These results suggest that such arousal was most likely due to human's anxiety in response to approaching danger rather than general excitement caused by watching flying robots. Moreover, analysis of the simultaneous head motion showed that as the EDA signal increased with the approaching drone, people made characteristic collision-avoidance movements by jerking their heads away from the drone. These findings again suggest that the physiological arousal signals observed in our study are most likely a result of people's anxiety to avoid impending danger rather than general excitement. Thus, in the following sections, we consider the EDA signal as an operational approximation of the human's perceived safety for the optimal path generation algorithm.

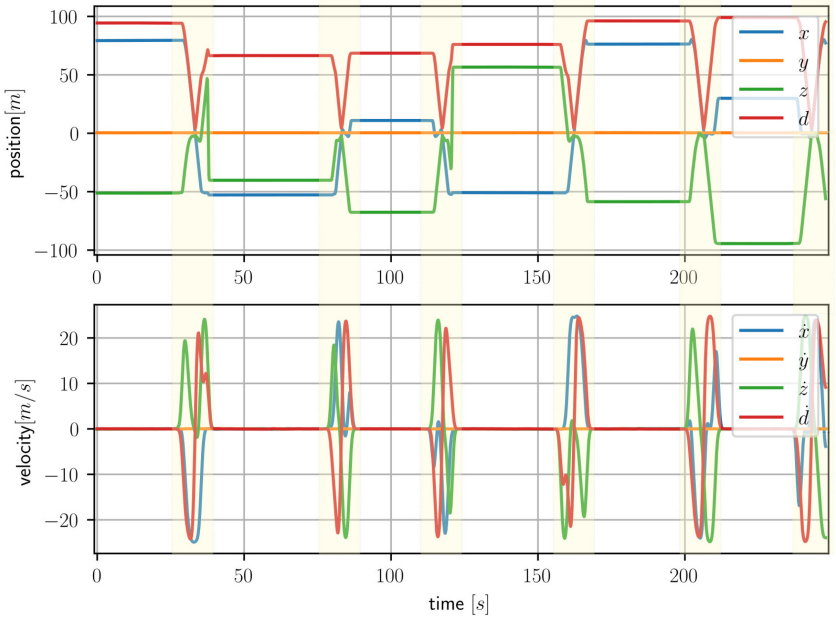


Fig. 4. Human-aerial robot interaction events. Test events and flight paths, x, y, z, denote position coordinates, and d denotes the distance between the robot and the human. Also, $\dot{x}$, $\dot{y}$, $\dot{z}$ denote velocity coordinates, and $\dot{d}$ denotes the rate of change of the distance.

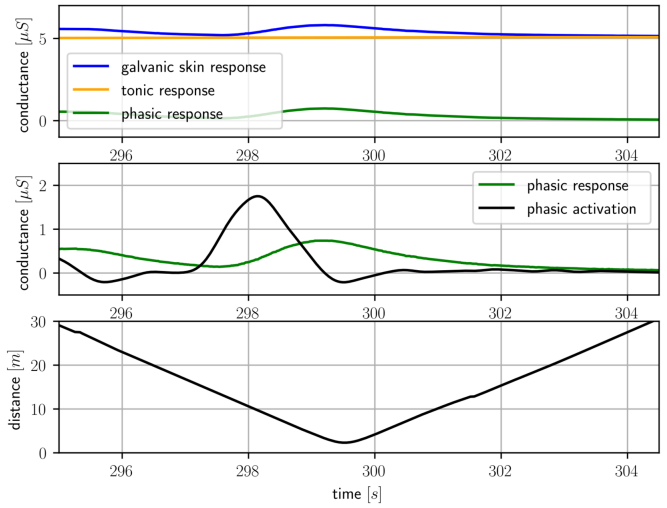


Fig. 5. EDA analysis result (phasic/tonic decomposition and deconvolution to determine phasic activation).

3 PROPOSED MODEL

We aim to develop a data-driven model that predicts the phasic activation (arousal), given the robot's position and velocity. Let $y_n \in \mathbb{R}$ denote the phasic activation, where n is the time index. The input (feature) variable denoted by $x_n \in \mathbb{R}^8$ is the vector that contains the distance to the robot, the rate of change of the distance, the Cartesian position coordinates and the velocity coordinates.

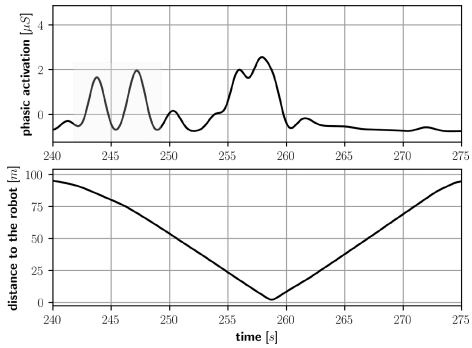


Fig. 6. Phasic activation signal induced by the flying robot. The shaded box indicates the response when the robot is in the far distance (greater than 60 [m]).

The challenge to be considered in the model is the uncertainties due to unknown factors. Despite the high-fidelity test environment, it is impossible to measure every stimulus on the subject, i.e., there are unknown factors in the data. As an example, one of our collected datasets, shown in Figure 6, illustrates the unknown factors present in the data. One can notice an increase of the phasic activation in the shaded area, although the flying robot is far away and virtually invisible to the subject.

To consider the unknown factors in the data, we would like to think that the unexpected spike of the phasic activation in Figure 6 is due to the change of the participant's focus of attention, i.e., the participant is distracted by some other stimulus. Inspired by the work in Mozer et al. (2007), we model the sequential dependence of human's focus of attention using a hidden Markov model (HMM). The HMM has two states represented by a latent variable, $z_n \in \{1, 2\}$, which represents the human's attention state, i.e.,

$$z_n := \begin{cases} 1, & \text{if the human is attentive to the robot,} \\ 2, & \text{otherwise.} \end{cases}$$

Then z_n is modeled by a homogeneous Markov chain with the following probability transition equation:

$$\pi_{n+1} = \pi_n \mathbf{A}. \tag{1}$$

The vector $\pi_k := [p(z_n = 1), p(z_n = 2)]$ is the stochastic row vector for the distribution over the state z_n, and $\mathbf{A} \in \mathbb{R}^{2\times 2}$ denotes the transition probability matrix of the Markov chain.[3] The initial distribution π_1 and $\mathbf{A}$ are the parameters of the Markov chain.

The attention state latent variable z_n assigns one of the two output emission models $f_\beta(x_n) + \epsilon$ or an independent random source δ as follows:

$$y_n = \mathbb{1}_{\{z_n=1\}}(f_\beta(x_n) + \epsilon) + \mathbb{1}_{\{z_n=2\}}\delta, \tag{2}$$

where $\mathbb{1}_A$ denotes the indicator function, and $f_\beta : \mathbb{R}^8 \to \mathbb{R}$ is a linear function $f_\beta(x) := \beta^\top \phi(x)$ with parameter β and basis[4] $\phi(x)$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$, and δ denotes the random source. As seen in Equation (2), y_n depends on x_n when $z_n = 1$; however, $y_n = \delta$ when $z_n = 2$, i.e., it is modeled as an independent random signal. In Equation (2), it can be seen that two regression functions of the model $y_n = f_\beta(x_n)$ and $y_n = \delta$ compete with each other to be chosen as the describer for the corresponding observation (x_n, y_n).

[3]We used $p(\cdot)$ for both probability and probability density; its distinction easily follows from the context.
[4]The third-order polynomial basis functions were chosen for the $\phi(\cdot)$.

ALGORITHM 1: EM Algorithm for MLE of θ given the data $(\mathbf{x}, \mathbf{y})$

Initialize the parameter, θ^{old} with θ^0.
repeat
1. Determine the **posterior**, $p(\mathbf{z}, \mathbf{w}|\mathbf{x}, \mathbf{y}, \theta^{old})$.
2. Calculate $Q(\theta, \theta^{old})$:

$$Q(\theta, \theta^{old}) := \sum_{\mathbf{z},\mathbf{w}} p(\mathbf{z}, \mathbf{w}|\mathbf{x}, \mathbf{y}, \theta^{old}) \log p(\mathbf{y}, \mathbf{z}, \mathbf{w}|\mathbf{x}, \theta),$$

which is the **expectation** of $\log p(\mathbf{y}, \mathbf{z}, \mathbf{w}|\mathbf{x}, \theta)$ with respect to the posterior.
3. Find the **maximizer**, θ^*

$$\theta^* = \operatorname*{argmax}_{\theta} Q(\theta, \theta^{old}),$$

4. Update θ^{old} with θ^*.
until *convergence*;

Selecting a model for δ with appropriate complexity is desired, otherwise only $y_n = f_\beta(x_n)$ would be active most of the time. We employ a mixture of Gaussians to model δ. The Gaussian mixture model (GMM) allows multi-modal and skewed distributions in contrast to the Gaussian distributions. The density of the mixture model for δ is defined by another latent variable $w_n \in \{1, \ldots, K\}$ as follows:

$$p(\delta|w_n = k) = \mathcal{N}\left(\delta|\mu_k, \sigma_k^2\right), \quad p(w_n = k) = \phi_k, \tag{3}$$

where ϕ_k are mixing coefficients such that $\sum_{k=1}^K \phi_k = 1$, $\mathcal{N}(\delta|\mu, \sigma)$ denotes a Gaussian density function of δ with the mean μ and the variance σ^2. We assume that w_n is independent and identically distributed. Also, it is assumed that w_n and z_n are independent, and furthermore, w_n and z_n are conditionally independent, given the observation (x_n, y_n).

3.1 Model Parameter Estimation

The model defined by Equations (1)–(3) has a set of parameters denoted by $\theta := \{\beta, \mu, \sigma, \pi_1, \mathbf{A}, \{\phi_i, \mu_i, \sigma_i\}_{i=1}^K\}$. Given the dataset $\mathbf{x} := \{x_1, \ldots, x_N\}$ and $\mathbf{y} := \{y_1, \ldots, y_N\}$, the parameter of the model is estimated by the maximum likelihood estimation (MLE) through the following conditional likelihood equation:

$$\operatorname*{argmax}_{\theta} p(\mathbf{y}|\mathbf{x}, \theta) = \operatorname*{argmax}_{\theta} \sum_{\mathbf{z}} \sum_{\mathbf{w}} p(\mathbf{y}, \mathbf{z}, \mathbf{w}|\mathbf{x}, \theta), \tag{4}$$

where the summation takes place over all possible sequences, $\mathbf{z} := \{z_1, \ldots, z_N\}$ and $\mathbf{w} := \{w_1, \ldots, w_N\}$. The number of terms for the summation is $2^N K^N$, which makes the optimization intractable for a large number of samples. Due to this challenge, Expectation-Maximization (EM) algorithm (Dempster et al. 1977) is widely used to obtain the MLE for HMM.

EM Algorithm: Let $(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{w})$ denote the complete data. Using the independence assumption on w_n, z_n and the Markov chain property as defined in Equation (1), the conditional likelihood for the complete data is calculated as follows:

$$p(\mathbf{y}, \mathbf{z}, \mathbf{w}|\mathbf{x}, \theta) = p(z_1|\pi_1) \left[\prod_{n=2}^N p(z_n|z_{n-1}, \mathbf{A})\right] \prod_{n=1}^N p(w_n|\theta) p(y_n|z_n, w_n, x_n, \theta), \tag{5}$$

where $\mathbf{z} := \{z_1, \ldots, z_N\}$ and $\mathbf{w} := \{w_1, \ldots, w_N\}$. The EM algorithm iteratively maximizes the likelihood in Equation (5) using only $(\mathbf{x}, \mathbf{y})$ as summarized in Algorithm 1.

3.2 The Calculation of the EM Algorithm for the Proposed Model

The EM algorithm for the proposed model is calculated by the following three subsequent steps:

Step 1. Determine the Posterior, $p(\mathbf{z}, \mathbf{w}|\mathbf{x}, \mathbf{y}, \theta^{old})$. Using the conditional independence assumption of the latent variables, the posterior is factorized as

$$p(\mathbf{z}, \mathbf{w}|\mathbf{x}, \mathbf{y}, \theta^{old}) = p(\mathbf{z}|\mathbf{x}, \mathbf{y}, \theta^{old})p(\mathbf{w}|\mathbf{x}, \mathbf{y}, \theta^{old}). \tag{6}$$

First, we calculate $p(\mathbf{z}|\mathbf{x}, \mathbf{y}, \theta^{old})$ using the *Forward-Backward Algorithm* (Baum et al. 1970). Define $a(z_{n,i})$ and $b(z_{n,i})$ as follows:

$$a(z_{n,i}) := p(y_1, \ldots, y_n, z_{n,i}),$$
$$b(z_{n,i}) := p(y_{n+1}, \ldots, y_N|z_{n,i}, \mathbf{x}),$$

where $z_{n,i}$ denotes the event $\{z_n = i\}$. To calculate the posterior, the following recursive equations are used:

$$a(z_{n,i}) = p(y_n|z_{n,i}, \mathbf{x}, \theta^{old}) \sum_{k=1}^{2} a(z_{n-1,k})p(z_n|z_{n-1,k}, \mathbf{A}^{old}),$$

$$b(z_{n,i}) = \sum_{k=1}^{2} b(z_{n+1,k})p(y_{n+1}|z_{n+1,k}, \mathbf{x}, \theta^{old})p(z_{n+1,k}|z_{n,i}, \mathbf{A}^{old}),$$

where $p(y_n|z_{n,i}, \mathbf{x}, \theta^{old})$ is calculated using Equation (2) as

$$p(y_n|z_{n,i}, \mathbf{x}, \theta) = \begin{cases} \mathcal{N}(y_n - f_\beta(x_n)|0,\ \sigma^2), & \text{if } i = 1 \\ \sum_{k=1}^{K} \phi_k \mathcal{N}(y_n|\mu_k,\ \sigma_k^2), & \text{if } i = 2. \end{cases}$$

The boundary values $a(z_1)$ and $b(z_N)$ are determined as $a(z_1) = p(z_1|\pi_1)p(y_1|\mathbf{x}, \theta^{old})$ and $b(z_N) = 1$. After calculating $a(z_n)$ recursively and $b(z_n)$, the posterior is determined as follows:

$$p(z_n|\mathbf{x}, \mathbf{y}, \theta^{old}) = \frac{a(z_n)b(z_n)}{p(\mathbf{y}|\mathbf{x}, \theta^{old})}, \tag{7}$$

$$p(z_{n-1,j}, z_{n,k}|\mathbf{x}, \mathbf{y}, \theta^{old}) = \frac{a(z_{n-1,j})p(y_n|z_{n,k}, x_n, \theta^{old})A_{jk}b(z_{n,k})}{p(\mathbf{y}|\mathbf{x}, \theta^{old})}, \tag{8}$$

where the likelihood is calculated as

$$p(\mathbf{y}|\mathbf{x}, \theta^{old}) = \sum_{k=1}^{2} a(z_{N,k}). \tag{9}$$

We calculate the posterior $p(\mathbf{w}|\mathbf{x}, \mathbf{y}, \theta^{old})$ independently from $p(z_n|\mathbf{x}, \mathbf{y}, \theta^{old})$ due to the conditional independence assumption of w_n and z_n:

$$p(w_{n,i}|x_n, y_n, \theta^{old}) = \frac{\phi_i \mathcal{N}(y_n|\mu_i, \sigma_i^2)}{\sum_{k=1}^{K} \phi_k \mathcal{N}(y_n|\mu_k, \sigma_k^2)}, \tag{10}$$

where $w_{n,i}$ denotes the event $\{w_n = i\}$.

Step 2. Calculate $Q(\theta, \theta^{old})$. Using the posterior calculated in Equations (7) and (10) and the likelihood in Equation (5), $Q(\theta, \theta^{old})$, defined in Algorithm 1, is calculated by expanding the log term:

$$\begin{aligned} &Q(\theta,\theta^{old}) \\ &:= \sum_{z,w} p(\mathbf{z},\mathbf{w}|\mathbf{x},\mathbf{y},\theta^{old}) \log p(\mathbf{y},\mathbf{z},\mathbf{w}|\mathbf{x},\theta) \\ &= \sum_{i=1}^{2} p(z_{1,i}|\mathbf{x},\mathbf{y},\theta^{old}) \log \pi_{1,i} \\ &\quad + \sum_{n=2}^{N}\sum_{i=1}^{2}\sum_{j=1}^{2} p(z_{n-1,i}, z_{n,j}|\mathbf{x},\mathbf{y},\theta^{old}) \log A_{i,j} \\ &\quad + \sum_{n=1}^{N}\sum_{i=1}^{2}\sum_{k=1}^{K} p(z_{n,i}, w_{n,k}|\mathbf{x},\mathbf{y},\theta^{old}) \log p(w_n|\phi_k) p(y_n|z_{n,i}, w_{n,k}, x_n, \theta), \end{aligned} \tag{11}$$

where $z_{n,i}$ denotes the event $\{z_n = i\}$, $w_{n,k}$ denotes the event $\{w_n = k\}$, $A_{i,j}$ is the (i,j) element of the matrix $\mathbf{A}$, and $p(y_n|z_{n,i}, w_{n,k}, x_n, \theta)$ is calculated using the model Equation (2) as follows:

$$\begin{aligned} &p(y_n|z_{n,i}, w_{n,k}, x_n, \theta) \\ &= \begin{cases} \mathcal{N}(y_n - f_\beta(x_n)|\mu,\ \sigma^2), & \text{if } i = 1 \\ \mathcal{N}(y_n|\mu_k,\ \sigma_k^2), & \text{if } i = 2. \end{cases} \end{aligned} \tag{12}$$

Step 3. Find Maximizer θ^*. As seen in Equation (11), each term has a distinct set of parameters. Hence, we can determine the maximizer for each term independently from the other terms. Using the constraints[5] of the probability distribution $\pi_{1,i}$ and the probability matrix $A_{i,j}$, the KKT (Karush-Kuhn-Tucker) condition (Bertsekas 1999, pp. 315) determines the maximizer as follows:

$$\pi^*_{1,i} = \frac{p(z_{1,i}|\mathbf{x},\mathbf{y},\theta^{old})}{\sum_{j=1}^{2} p(z_{1,j}|\mathbf{x},\mathbf{y},\theta^{old})}, \tag{13}$$

$$A^*_{j,k} = \frac{\sum_{n=2}^{N} p(z_{n-1,j}, z_{n,k}|\mathbf{x},\mathbf{y},\theta^{old})}{\sum_{l=1}^{2}\sum_{n=2}^{N} p(z_{n-1,j}, z_{n,l}|\mathbf{x},\mathbf{y},\theta^{old})}. \tag{14}$$

The maximizer for the last term in Equation (11) is calculated using the model Equation (2). Let L denote the last term in Equation (11). Using Equation (12), L is written as follows:

$$\begin{aligned} &L(\beta, \sigma, \{\phi_i, \mu_i, \sigma_i\}_{i=1}^K) \\ &= \sum_{n=1}^{N}\sum_{i=1}^{2}\sum_{k=1}^{K} p(z_{n,i}, w_{n,k}|\mathbf{x},\mathbf{y},\theta^{old}) \log p(w_n|\phi_k) p(y_n|z_{n,i}, w_{n,k}, x_n, \theta) \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} p(w_{n,k}|\mathbf{x},\mathbf{y},\theta^{old}) \log \phi_k \\ &\quad + \sum_{n=1}^{N} p(z_{n,1}|\mathbf{x},\mathbf{y},\theta^{old}) \log\left(\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y_n - f_\beta(x_n))^2}{2\sigma^2}\right)\right) \\ &\quad + \sum_{n=1}^{N}\sum_{k=1}^{K} p(z_{n,2}, w_{n,k}|\mathbf{x},\mathbf{y},\theta^{old}) \log\left(\frac{1}{\sigma_k\sqrt{2\pi}} \exp\left(-\frac{(y_n - \mu_k)^2}{2\sigma_k^2}\right)\right). \end{aligned}$$

[5]The sum of the probability distribution equals to one and the probability matrix's row-wise sums equal to one.

The variable $\beta^* := \text{argmax}_\beta Q(\beta, \theta^{old})$ is calculated as

$$\beta^* := \text{argmin}_\beta \sum_{n=1}^{N} p(z_{n,1}|\mathbf{x}, \mathbf{y}, \theta^{old})(y_n - f_\beta(x_n))^2. \tag{15}$$

The ϕ_i^*, μ_i^*, σ_i^*, and σ^* are determined using KKT (Karush-Kuhn-Tucker) condition as follows:

$$\phi_i^* = \frac{\sum_{n=1}^{N} p(w_{n,i}|\mathbf{x}, \mathbf{y}, \theta^{old})}{\sum_{n=1}^{N}\sum_{k=1}^{K} p(w_{n,k}|\mathbf{x}, \mathbf{y}, \theta^{old})},$$

$$\mu_i^* = \frac{\sum_{n=1}^{N} p(z_{n,2}, w_{n,i}|\mathbf{x}, \mathbf{y}, \theta^{old})\, y_n}{\sum_{n=1}^{N} p(z_{n,2}, w_{n,i}|\mathbf{x}, \mathbf{y}, \theta^{old})},$$

$$\sigma_i^* = \frac{\sum_{n=1}^{N} p(z_{n,2}, w_{n,i}|\mathbf{x}, \mathbf{y}, \theta^{old})\, (y_n - \mu_i^*)^2}{\sum_{n=1}^{N} p(z_{n,2}, w_{n,i}|\mathbf{x}, \mathbf{y}, \theta^{old})},$$

$$\sigma^* = \frac{\sum_{n=1}^{N} p(z_{n,1}|\mathbf{x}, \mathbf{y}, \theta^{old})\, (y_n - f_{\beta^*}(x_n))^2}{\sum_{n=1}^{N} p(z_{n,1}|\mathbf{x}, \mathbf{y}, \theta^{old})}.$$

Remark 1. In contrast to the mean-squared error minimization, the latent variable model (HMM) determines the parameter of $f_\beta(x)$ as the weighted least squared error solution with the weight of the posterior $p(z_{n,1}|\mathbf{x}, \mathbf{y}, \theta^{old})$ as follows:

$$\beta^* := \text{argmin}_\beta \sum_{n=1}^{N} p(z_{n,1}|\mathbf{x}, \mathbf{y}, \theta^{old})(y_n - f_\beta(x_n))^2,$$

where $p(z_{n,1}|\mathbf{x}, \mathbf{y}, \theta^{old})$ denotes $p(z_n = 1|\mathbf{x}, \mathbf{y}, \theta^{old})$. Unlike the Gaussian noise model, the proposed method puts greater weight on the samples, which are more relevant to the input based on the posterior of the attention state.

3.3 Result

We choose the following initial parameter θ^0:

(1) $\beta^0 := \text{argmin}_\beta \sum_{n=1}^{N}(y_n - f_\beta(x_n))^2$, and $\sigma^0 = 0.5$,
(2) $\mathbf{A}^0 := \left[\begin{smallmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{smallmatrix}\right]$, and $\pi_1^0 := [1/2, 1/2]$,
(3) $\phi_k^0 = 1/K$, $\sigma_k^0 = 1$, and μ_k are randomly chosen from the interval $[-1, 1]$,

where K is the number of the Gaussian basis of the GMM. For the linear function $f_\beta(x) := \beta^\top \phi(x)$, we choose the basis functions $\phi(x)$ as polynomials with degree 3.

The Gaussian i.i.d. noise model (which minimizes MSE) is contained in the proposed model structure as a special case, $p(z_n = 1) = 1$, i.e., the arousal is always better explained by $f_\beta(x)$ than the random source δ. Since HMM is non-identifiable, in general, the likelihood-ratio model comparison test with the *training dataset* does not apply. To determine which model is more suitable, we calculate the likelihood with *test dataset* by employing the approach in Uria et al. (2016). We randomly partition the data from 56 subjects into a training set with 38 subjects and a test set with the other 18 subjects. Figure 7 shows the log-likelihood with test data set for the models: (i) the Gaussian i.i.d. noise model, (ii) the proposed HMM model with a different number of basis K for the GMM. The significant increase in likelihood using the proposed model as compared to the Gaussian i.i.d. noise model suggests that the proposed model is more suitable for the experimental data at hand than the Gaussian i.i.d. noise model.

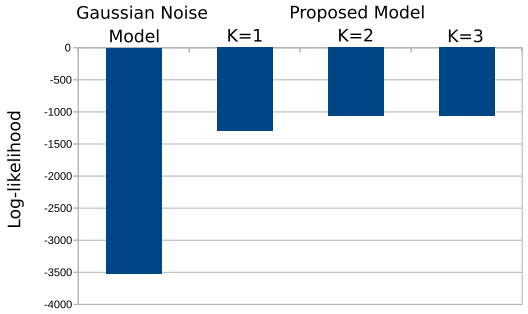


Fig. 7. Log-likelihood with test data. Gaussian noise model refers to $y_n = f_\beta(x_n) + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and K denotes the number of Gaussian basis in the Equation (3).

Remark 2. The improvement of the log-likelihood by using the proposed model suggests that the null hypothesis[6] is not true. By rejecting the null hypothesis, we show that the proposed model is more suitable than the Gaussian noise model. Consider the following log-likelihood ratio test (see 11.7.4 in Wasserman (2013)):

$$H_0 : \varphi \in \Theta_0 \quad \text{versus} \quad H_1 : \varphi \in \Theta_1,$$

where φ denotes the true parameter, Θ_0 denotes the set of parameters for the Gaussian noise model, and Θ_1 denotes the set of parameters for the proposed model. Note that the Gaussian noise model is a special case of the proposed model, i.e., $\Theta_0 \subseteq \Theta_1$. The likelihood ratio statistics λ is calculated as

$$\lambda = 2 \log \left(\frac{\sup_{\theta \in \Theta_1} \mathcal{L}(\theta)}{\sup_{\theta \in \Theta_0} \mathcal{L}(\theta)} \right),$$

where $\mathcal{L}(\theta) := p(\mathbf{y}, \mathbf{z}, \mathbf{w} | \mathbf{x}, \theta)$ denotes the likelihood for θ in Equation (5). Figure 7 shows that $\lambda > 4{,}000$. The relative degree is $r = 10$, as the proposed model has 10 more parameters than the Gaussian noise model. The likelihood ratio test is: reject H_0, when $\lambda > \chi^2_{r,\alpha}$. We reject $H_0 : \varphi \in \Theta_0$ with p-value at 0.01.

We fix the proposed model with $K = 2$, since a greater number of basis does not result in significant improvement in likelihood as shown in Figure 7. The function f_β with the fixed model is used to predict the phasic activation (arousal) as shown in Figure 8. Figure 9 shows that the MSE minimization model (the Gaussian noise model) has few signs of over-fitting (oscillation and spiky shape). In the following section, optimal path planning with the two prediction models is analyzed.

4 OPTIMAL PATH PLANNING

In this section, we formulate the optimal path planning as an optimal control problem. The assumptions on the vehicle's dynamics, constraints on acceleration, and velocity are introduced. Subsequently, we present the collision avoidance requirements and an optimal control formulation. We assume the co-robot navigates on the 2D plane. The vehicle is assumed to be a double integrator with dynamics:

$$\begin{aligned} \frac{d^2}{dt^2}\mathbf{p}(t) &= \mathbf{u}(t), \\ \mathbf{p}(0) = \mathbf{p}_{\text{in}}, &\quad \dot{\mathbf{p}}(0) = \mathbf{v}_{\text{in}}, \end{aligned} \tag{16}$$

[6]The null hypothesis assumes that the true parameter φ is in the set of parameters Θ_0, which corresponds to the Gaussian i.i.d. noise model.

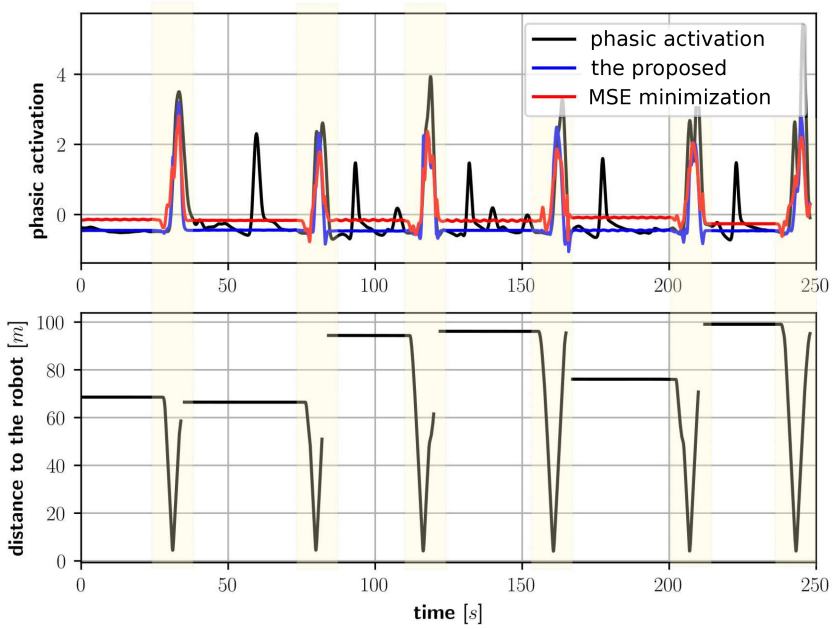


Fig. 8. Prediction, $\hat{\mathbf{y}} = f_\beta(\mathbf{x})$.

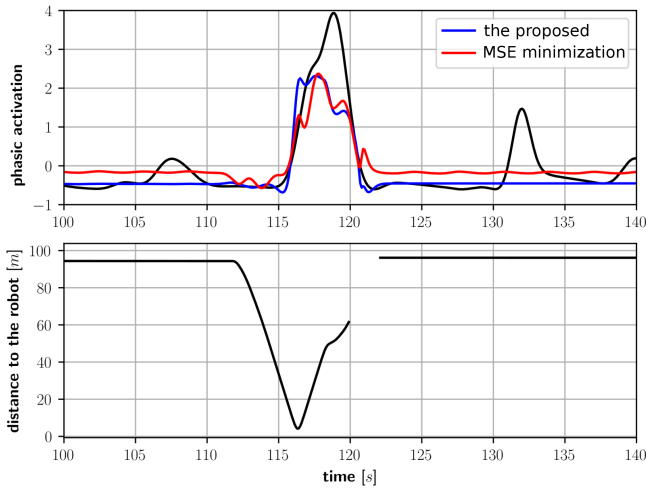


Fig. 9. Closer look of the prediction on an event.

where $\mathbf{p}(t) := [x(t), y(t)]^\top$ is the planar position vector, $\mathbf{p}_{\text{in}}$ denotes initial position, and $\mathbf{v}_{\text{in}}$ denotes initial velocity.

The proposed cost function must account for the relative positions and velocities with respect to the human in the environment. The human is modeled as a stationary object. The associated cost function takes the following form:

$$J = \int_0^{t_f} l(\mathbf{p}(t), \dot{\mathbf{p}}(t), \mathbf{h}) dt, \tag{17}$$

where $l(\cdot)$ denotes a running cost function, $\mathbf{h}$ denotes the position of the human, and t_f is the total flight time.

Remark 3. In the previous section, we estimated the function that maps the robot's position and velocity to the phasic activation of EDA in the form of the parametric function $f_\beta(\cdot)$ that can be found in Equation (2). We will consider $f_\beta(\cdot)$ in the cost function J as a constraint using the penalty method as in Equation (28) for the optimal path planning described in Section 4.3.

The collision avoidance constraint is defined with a lower bound on the spatial separation from the human as

$$\|\mathbf{p}(t) - \mathbf{h}\|_2 \geq d_s, \quad \forall t \in [0, t_f], \tag{18}$$

where the human is modeled as a disk with centers at $\mathbf{h}$, and d_s is the safe distance from the center of human to the robot. The safe distance is a design parameter that depends on how close the robot is allowed to approach the humans, regardless of the velocity.

As previously mentioned, our goal is not only to optimize for the perceived safety of individuals but to generate trajectories that the co-robots can follow. Therefore, these trajectories must not only satisfy the dynamics in Equation (16) but also respect the physical limitations of the vehicles. Thus, we define the bound b_v on the velocity and b_a on the acceleration of the flying robot:

$$|\dot{x}(t)| \leq b_v, \quad |\dot{y}(t)| \leq b_v, \quad |\ddot{x}(t)| \leq b_a, \quad |\ddot{y}(t)| \leq b_a, \quad \forall t \in [0, t_f]. \tag{19}$$

When generating trajectories for autonomous vehicles, we desire to achieve a certain final position (with a chosen final velocity) that is predetermined by a higher level task. Additionally, the maximum allowable time must be prespecified to ensure that the task is executed in a realistic time frame. These boundary conditions are

$$\mathbf{p}(t_f) = \mathbf{p}_{\text{term}}, \quad \dot{\mathbf{p}}(t_f) = \mathbf{v}_{\text{term}}, \quad t_{\min} \leq t_f \leq t_{\max}, \tag{20}$$

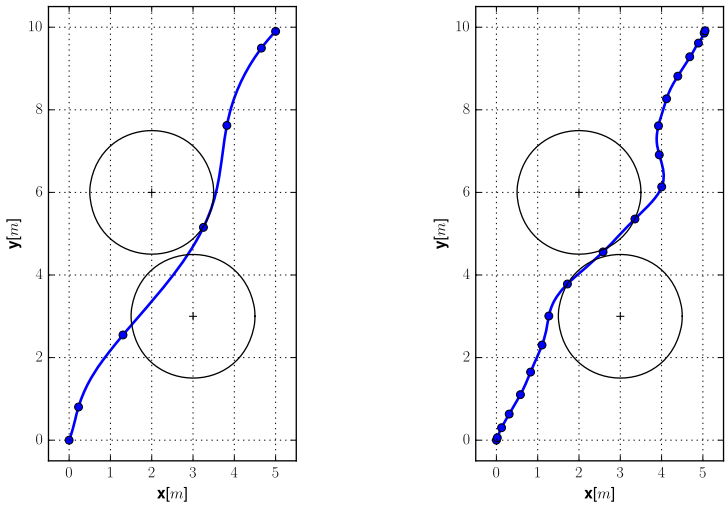
where $\mathbf{p}_{\text{term}}$ denotes the terminal position at the target, $\mathbf{v}_{\text{term}}$ denotes the terminal velocity, and $t_{\min}$, $t_{\max}$ are the bounds on the flight time.

Given the prior formulation, the proposed trajectory generation method can be cast into a fixed-end-point optimal control problem, where the desired trajectory is obtained by minimizing the cost J defined in Equation (17) and satisfying the constraints as follows:

$$\begin{aligned} &\min_{\mathbf{x}(t), t_f} J \\ &\text{subject to collision avoidance constraint in Equation (18),} \\ &\qquad\qquad \text{dynamic constraints of the robot in Equation (19),} \\ &\qquad\qquad \text{boundary conditon in Equation (20).} \end{aligned}$$

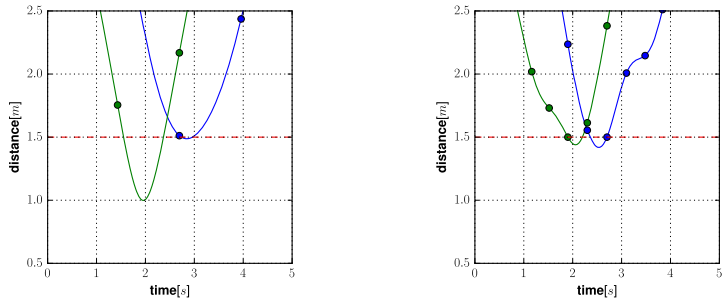
4.1 Finite-dimensional Approximation for Optimal Trajectory Generation

This section introduces the concurrent parameterizations of the trajectories in a finite-dimensional space. To leverage the advantages of algorithms for specific polynomial structures, we introduce two different polynomial basis functions that describe the same curve. This framework can account for a large class of curves, because a polynomial function can uniformly approximate any continuous function with an arbitrarily small error (Weierstrass approximation theorem) (Stone 1948). The PS method relies on the discretization of the polynomial to find the optimized trajectory. This results in a drawback for safety-critical systems, because the constraints are only verified for the finite number of discretized time nodes, as discussed in Choe et al. (2013). This phenomena is illustrated in Figure 10, where none of the *discretization points* violate the constraints from Equation (18). The resulting trajectory violates the collision avoidance objective. Furthermore,



(a) number of nodes: 7 (b) number of nodes: 20

Fig. 10. Trajectories generated by PS method to minimize flight time with different numbers of time nodes.



(a) number of nodes: 7 (b) number of nodes: 20

Fig. 11. Distance from the centers of circles to the trajectories by PS method.

Figure 11 shows that by increasing the number of nodes, the PS method does not necessarily provide a collision-free path. To overcome this issue, we propose a Bézier curve (Farin and Hansford 2000) as a second way to represent the flight path.

4.1.1 Two Representations of a Polynomial Trajectory. From this moment forward, we will always consider the co-robot's trajectory $\mathbf{p}(t) := [x(t), y(t)]^\top$ as an nth order polynomial in $x(t)$ and $y(t)$. We present a brief overview of both equivalent representations.

(1) A degree n Bézier curve is given by

$$\mathbf{p}(t) = \sum_{k=0}^{n} \bar{\mathbf{x}}_k b_k^n(\zeta(t)), \quad \zeta : [0, t_f] \to [0, 1], \quad \zeta(t) := \frac{t}{t_f}, \tag{21}$$

where

$$b_k^n(\zeta) := \binom{n}{k}(1-\zeta)^{n-k}\zeta^k, \quad \zeta \in [0, 1]$$

are the Bernstein polynomial basis, and the coefficients $\bar{\mathbf{x}}_k \in \mathbb{R}^2$ are called control points of the Bézier curve.

(2) The interpolation curve representation is as

$$\mathbf{p}(t) = \sum_{k=0}^{n} \mathbf{x}_k \ell_k(\eta(t)), \quad \eta : [0, t_f] \to [-1, 1], \quad \eta(t) := \frac{2t}{t_f} - 1, \tag{22}$$

where

$$\ell_k(\eta(t)) := \prod_{0 \le i \le n, i \ne k} \left(\frac{\eta(t) - \eta(t_i)}{\eta(t_k) - \eta(t_i)} \right)$$

are the Lagrange polynomial basis and $\mathbf{x}_k \in \mathbb{R}^2$ are interpolation points at time nodes t_k.

4.1.2 Cost Function Calculation with Interpolation Curve and Legendre-Gauss-Lobatto (LGL) Quadrature. We can use the interpolation curve representation with LGL nodes to calculate the cost function. First, the interpolation points are determined on the nodes, and then LGL quadrature is applied. The procedure is written in the following steps:

STEP 1. Determine Interpolation Points. The LGL nodes are determined by finding the roots of the first derivative of the nth order Legendre polynomial $P_n(\eta)$, i.e., the set of LGL nodes $\{\eta_k\}$ is $\{-1, 1\} \cup \{\eta \mid P_n'(\eta) = 0\}$. Using the inverse map $\eta^{-1} : [-1, 1] \to [0, t_f]$, the set of LGL nodes $\eta_k \in [-1, 1]$ is mapped to the time nodes $t_k \in [0, t_f]$. Substituting the time nodes t_k into the given trajectory $\mathbf{p}(t)$, the interpolation points are determined as $\mathbf{p}(t_k)$. Now the polynomial trajectory $\mathbf{p}(t)$ is represented as an interpolation curve with LGL nodes.

STEP 2. LGL quadrature (Hunter and Nikolov 2000). Let $f(t)$ denote the integrand of the cost function in Equation (17), i.e., $f(t) := l(\mathbf{p}(t), \dot{\mathbf{p}}(t), \mathbf{h})$ for fixed trajectory $\mathbf{x}(t)$ and parameters $\mathbf{h}$. The LGL quadrature calculates the value of the cost function J as follows:

$$J = \int_0^{t_f} f(t)dt = \frac{t_f}{2} \left[\frac{2(f(1) + f(-1))}{n(n+1)} + \sum_{k=2}^{n} w_k f(t_k) + R_n \right], \tag{23}$$

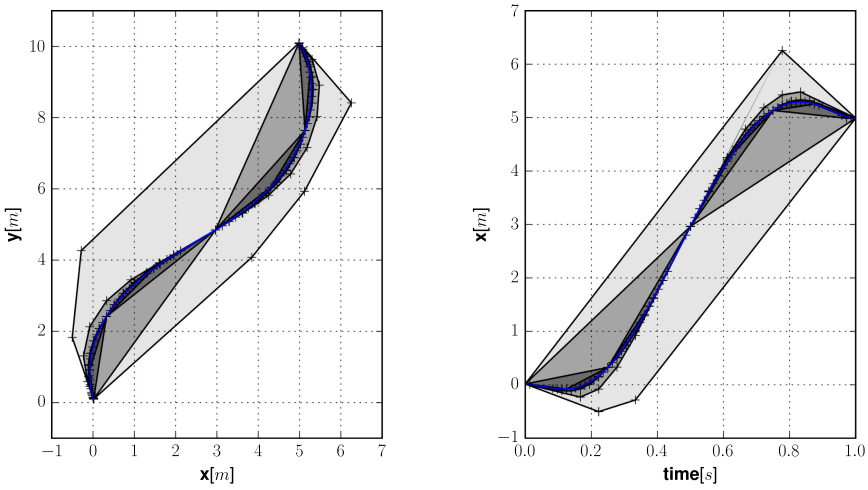
where the weight w_k and the remainder are

$$w_k = \frac{2}{n(n+1)P_n(\eta_k)},$$
$$R_n = \frac{-n^3(n+1)2^{2n+1}[(n-1)!]^4}{(2n+1)[2n!]^3} f^{(2n)}(\eta), \quad \text{for some } \eta \in (-1, 1),$$

and $f^{(2n)}(\cdot)$ denotes $2n$th derivative of the function $f(\cdot)$.

Remark 4. The remainder decays fast as n increases. For example, for 10th order polynomial trajectory, the LGL quadrature can calculate the cost function J with the error bound $|R_{10}| \le 1.5 \times 10^{-24} \sup_{\eta \in (-1,1)} |f^{(2n)}(\eta)|$.

4.1.3 Collision Avoidance Constraint with Bézier Curve Representation. The convex hull containment property of Bézier curve is used to verify the spatial constraints (Choe et al. 2013). As illustrated in Figure 12, the convex hulls contain the trajectories in the projected spaces of (x, y) and (t, x). Splitting the Bézier Curve with De Casteljau's algorithm (De Casteljau 1959; Farin and Hansford 2000), the convex hulls give tighter containment of the trajectory. The flatness factor of Bézier Curve defined in Equation (24) is used to determine when to stop splitting the curves. Decreasing the flatness factor, the control points move towards the polynomial path as illustrated in Figure 13:



(a) Trajectory on (x,y) plane. (b) Trajectory on (t,x) plane.

Fig. 12. Containment of trajectory in the convex hulls.

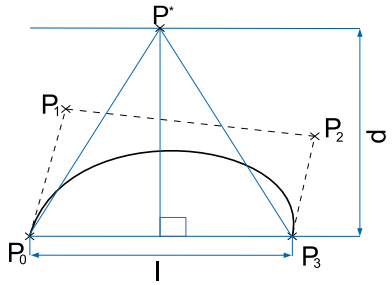


Fig. 13. Flatness factor. The equilateral triangle of $P_0P^*P_3$ has the same perimeter as the convex hull with vertices P_0, P_1, P_2, P_3.

$$\gamma(\{\bar{\mathbf{x}}_{(i,k)}\}) := \frac{\sum_{k=0}^{n-1} \|\bar{\mathbf{x}}_{(i,k)} - \bar{\mathbf{x}}_{(i,k+1)}\|_2}{\|\bar{\mathbf{x}}_{(i,0)} - \bar{\mathbf{x}}_{(i,n)}\|_2}. \tag{24}$$

Remark 5. Splitting the curve (blue) repeatedly into two halves using De Casteljau's algorithm (De Casteljau 1959; Farin and Hansford 2000), the convex hulls (gray) covering the curve become flatter, i.e., $\gamma \to 1$. Note that the containment also holds for the time and position coordinate plane.

Remark 6. Flatness factor γ for the control points P_0, P_1, P_2, P_3 is calculated as in Equation (24):

$$\gamma = \frac{|P_0P_1| + |P_1P_2| + |P_2P_3|}{|P_0P_3|},$$

where $|P_0P_1|$ denotes the distance between the control points P_0 and P_1. Using elementary calculations, the maximum possible distance of the control point given the flatness factor γ and the distance l from the starting point P_0 and the terminal point P_3 is calculated as

$$d = \frac{l}{2}\left(1 - \gamma^2\right).$$

So, having γ close to 1 brings the control points P_0, P_1, P_2, P_3 to the line connecting P_0 and P_3.

The distance between the convex hulls, denoted $\mathbf{C}_i$, which contain the trajectory and obstacles, denoted $\mathbf{O}_j$, is calculated by applying GJK algorithm (Gilbert et al. 1988). The procedure to determine a lower bound on the minimum distance between the flight path and obstacles is written in Algorithm 2.

ALGORITHM 2: Lower bound of the minimum distance

Step 1. Given the set of control points, $\{\bar{\mathbf{x}}_{(0,k)}\}$, of the trajectory $\mathbf{p}(t)$ split the trajectory into $\{\{\bar{\mathbf{x}}_{(0,k)}\}, \{\bar{\mathbf{x}}_{(1,k)}\}\}$ until each of the split curves is flat enough.

repeat
  for $\bar{\mathbf{x}}_{(i,k)}$ *in* $\{\bar{\mathbf{x}}_{(i,k)}\}$ **do**
    Calculate the flatness factor $\gamma(\{\bar{\mathbf{x}}_{(i,k)}\})$ according to (24).
    if $\gamma \geq$ *threshold* **then**
      Split the Bézier curve using De Casteljau algorithm.
    end
  end
until $\max\{\gamma(\bar{\mathbf{x}}_{(0,k)}), \gamma(\bar{\mathbf{x}}_{(1,k)}), \ldots, \gamma(\bar{\mathbf{x}}_{(N,k)})\} \leq$ *threshold*;

Step 2. From the split Bézier curves with the set of control points, $\{\{\bar{\mathbf{x}}_{(0,k)}\}, \{\bar{\mathbf{x}}_{(1,k)}\}, \ldots, \{\bar{\mathbf{x}}_{(N,k)}\}\}$, in Step 1, determine the distances between each convex hull of control points, $\{\bar{\mathbf{x}}_{(i,k)}\}$, and each obstacle, $\mathbf{O}_j$.

for a geometric obstacle, $\mathbf{O}_j$ **do**
  Calculate the distance, $d(\mathbf{C}_i, \mathbf{O}_j)$,between $\mathbf{C}_i$ and $\mathbf{O}_i$ with GJK algorithm.
end
return *the distance, d*

$$d(\mathbf{y}) := \min_{i,j} d(\mathbf{C}_i, \mathbf{O}_j)$$

4.1.4 Dynamic Constraints with Bézier Curve Representation. Constraints on velocity and acceleration can also be verified using the Bézier curve representation. Taking the time derivative on the trajectory $\mathbf{x}(t)$ in Equation (21), the velocity $\mathbf{v}(t)$ is written as a Bézier curve with the velocity control points $\bar{\mathbf{v}}_k$ as follows:

$$\mathbf{v}(t) = \sum_{k=0}^{n-1} \bar{\mathbf{v}}_k b_k^{n-1}(\zeta(t)), \quad \bar{\mathbf{v}}_k := \left(\frac{n}{t_f}\right)(\bar{\mathbf{x}}_{k+1} - \bar{\mathbf{x}}_k).$$

Similarly, the acceleration $\mathbf{a}(t)$ is written with the acceleration control points $\bar{\mathbf{a}}_k$ as follows:

$$\mathbf{a}(t) = \sum_{k=0}^{n-2} \bar{\mathbf{a}}_k b_k^{n-2}(\zeta(t)), \quad \bar{\mathbf{a}}_k := \frac{n(n-1)}{t_f^2}(\bar{\mathbf{x}}_{k+2} - 2\bar{\mathbf{x}}_{k+1} + \bar{\mathbf{x}}_k).$$

Since $\mathbf{v}(t)$ and $\mathbf{a}(t)$ are also Bézier curves, the convex hull of the control points contains $\mathbf{v}(t)$ and $\mathbf{a}(t)$, respectively. With a similar procedure used to calculate minimum distance bound between the trajectory and the obstacle in Algorithm 2, upper bounds on the magnitude of $\mathbf{v}(t)$ and $\mathbf{a}(t)$ are calculated.

Remark 7. Once a tight containment of the velocity curves is attained, the maximum value of velocity control points for each coordinate is an upper bound for the velocity. Through the same procedure, an upper bound for the acceleration is calculated. These upper bounds are used to check the constraints in Equation (20).

4.2 Finite-dimensional Optimal Trajectory Generation

Discretizing the optimal control problem with finite order polynomial approximation, the trajectory generation is cast as the following optimization problem. The decision variables $\mathbf{y}$ are the

control points and flight time, i.e., $\mathbf{y} = (\bar{\mathbf{x}}_0, \ldots, \bar{\mathbf{x}}_n, t_f)$, for an nth order polynomial trajectory with the terminal time t_f.

Cost function. From the LGL quadrature in Equation (23), ignoring the remainder term R_n, the cost function is written as follows:

$$F(\mathbf{y}) := \frac{t_f}{2}\left[\frac{2(f(1)+f(-1))}{n(n+1)} + \sum_{k=1}^{n-1} w_k f(t_k)\right]. \tag{25}$$

Constraints. Using Algorithm 2, the collision avoidance constraint, dynamic constraint, and boundary condition can be put together as the following constraint inequalities:

$$\mathbf{g}_l \le \mathbf{G}(\mathbf{y}) \le \mathbf{g}_u, \tag{26}$$

where

$$\mathbf{G}(\mathbf{y}) := \begin{bmatrix} d(\mathbf{y}) \\ b_v(\mathbf{y}) \\ b_a(\mathbf{y}) \\ t_f \\ \|\bar{\mathbf{x}}_0 - \mathbf{x}_{\text{in}}\|_2 \\ \|\bar{\mathbf{v}}_0 - \mathbf{v}_{\text{in}}\|_2 \\ \|\bar{\mathbf{x}}_n - \mathbf{x}_{\text{term}}\|_2 \\ \|\bar{\mathbf{v}}_n - \mathbf{v}_{\text{term}}\|_2 \end{bmatrix}, \quad \mathbf{g}_l := \begin{bmatrix} d_l \\ 0 \\ 0 \\ t_{\min} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{g}_u := \begin{bmatrix} \infty \\ b_v \\ b_a \\ t_{\max} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

d_l denotes the lower bound on the separation distance from the obstacles, b_v, b_a are the bounds on the velocity and acceleration, and $t_{\min}$, $t_{\max}$ are the bounds on the flight time. In the above constraint inequality Equation (26), the lower bound of separation distance from the obstacles for $\mathbf{y}$ denoted as $d(\mathbf{y})$ is calculated using Algorithm 2. The upper bounds on velocity and acceleration for $\mathbf{y}$, denoted as $b_v(\mathbf{y})$ and $b_a(\mathbf{y})$, respectively, are determined as described in Section 4.1.4.

The finite-dimensional nonlinear optimization problem to generate the trajectories is formulated as follows:

$$\min_{\mathbf{y}} F(\mathbf{y}), \tag{27}$$

subject to

$$\mathbf{g}_l \le \mathbf{G}(\mathbf{y}) \le \mathbf{g}_u.$$

The solution to the optimization problem can be obtained by employing standard optimization algorithms, such as interior point optimizer (IPOPT) (Wächter and Biegler 2006), sequential quadratic programming (SQP) (Bertsekas 1999, pp. 431), to name a few. A flight path that minimizes the flight time, while avoiding collisions, is presented as an example of the implementation of the optimal path planning in Figure 14.

4.3 Optimal Path Planning Considering the Perceived Safety Model

Define $\mathbf{x}(t)$ in the same way we defined x_n in Equation (2), where $\mathbf{x}(t) \in \mathbb{R}^8$ contains the distance to the robot, the rate of change of the distance, the position coordinates and the velocity coordinates at time t. Notice that with the polynomial path $\mathbf{p}(t)$ and $\dot{\mathbf{p}}(t)$ one can directly construct $\mathbf{x}(t)$. For this reason, to simplify the notation, we can use $\mathbf{x}(t)$ and $(\mathbf{p}(t), \dot{\mathbf{p}}(t))$ interchangeability as arguments of the functions $f_\beta(\cdot)$, $J(\cdot)$ and $l(\cdot)$.

In the optimal path planning, we only consider values of f_β larger than a threshold b_a, where $b_a \ge 0$ is essentially a tuning parameter. Intuitively, we ignore arousal levels below the threshold.

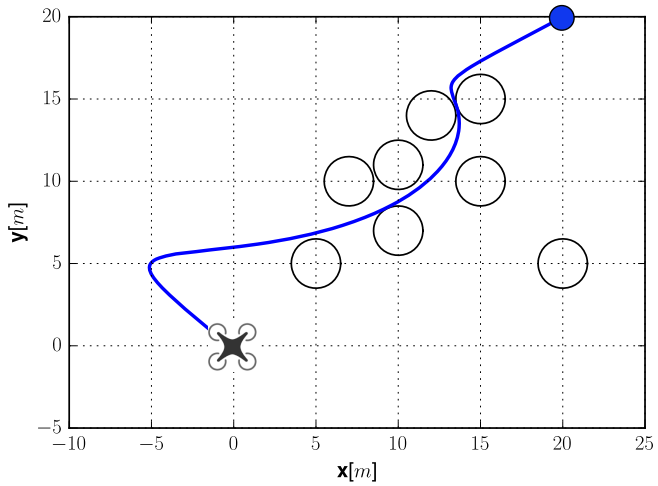


Fig. 14. An optimal flight path minimizing time while avoiding obstacles.

To make the optimization problem tractable, instead of adding a strict constraint, the constraint is incorporated in the running cost as a penalty function (Zangwill 1967):

$$l(\mathbf{p}(t), \dot{\mathbf{p}}(t), \mathbf{h}) := 1 + \gamma \max(0, f_\beta(\mathbf{x}(t)) - b_a)^2, \tag{28}$$

where γ is the penalty coefficient. The corresponding cost function $J(\bar{\mathbf{x}}_0, \ldots, \bar{\mathbf{x}}_n, t_f)$ becomes

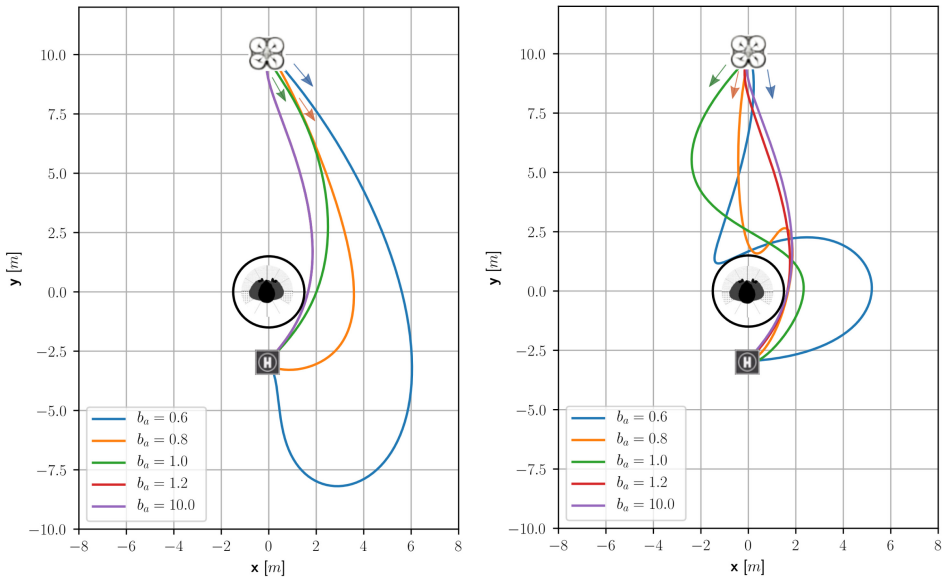
$$J(\bar{\mathbf{x}}_0, \ldots, \bar{\mathbf{x}}_n, t_f) = t_f + \gamma \int_0^{t_f} \max(0, f_\beta(\mathbf{x}(t)) - b_a)^2 dt. \tag{29}$$

Remark 8. Notice that this is fixed end-point calculus of variation problem, where the total flight time t_f is not fixed, yet $\mathbf{x}(t_f)$ is a fixed point, and we want to minimize the cost-function concurrently with the penalty term.

The two arousal prediction functions are used in the optimal flight trajectory generation, as shown in Figures 15(a) and 15(b). The smaller value of b_a results in a path that is more safety conscious, as intended by the running cost function in Equation (28). Flight paths generated with the proposed model show the desirable behavior, as shown in Figure 15(a) (decreasing b_a results in greater distance from the human). However, the paths with the Gaussian noise model have unconvincing shapes. This undesirable behavior of the Gaussian noise model (MSE minimization) is due to over-fitting, as we have seen in Figure 9. It shows that the standard regression (MSE minimization) model does not generalize in the optimal path generation task.

5 CONCLUSION

We present a path planning framework that takes into account the human's safety perception in the presence of a flying robot. We devise a machine learning method to estimate the uncertain parameters of the proposed safety perception model based on test data collected using Virtual Reality (VR) testbed. Also, an offline optimal control computation using the estimated safety perception model is presented. Due to the unknown factors in the human tests data, it is not suitable to use standard regression techniques that minimize the mean-squared error. We propose to use a Hidden Markov model (HMM) approach where human's attention is considered as a hidden state to infer whether the data samples are relevant to learn the safety perception model. The HMM approach



(a) The proposed HMM model. (b) The Gaussian i.i.d. noise model.

Fig. 15. Flight paths generated with the two perceived safety models. The circle around the human represents personal space, which the robot needs to avoid.

improved log-likelihood over the standard least squares solution. For path planning, we use Bernstein polynomials (Bézier curve) for discretization, as the resulting path remains within the convex hull of the control points, providing guarantees for deconfliction with obstacles at a low computational cost. An example of an optimal trajectory generation using the learned human model is presented. The optimal trajectory generated using the proposed model results in a reasonable safe distance from the human. In contrast, the paths generated using the standard regression model have undesirable shapes due to overfitting. The example demonstrates that the HMM approach has the robustness to the unknown factors compared to the standard MSE model.

5.1 Future Work and Discussion

A drawback of the proposed framework is the lack of adaptability, because the algorithms for estimation and path planning are off-line algorithms. For example, the pool of 56 subjects is probably not enough to generalize the arousal prediction model to many different scenarios. In contrast, an online algorithm can continue to update the parameter of the prediction model using the current data. Also, the off-line trajectory generation might not be suitable when the environment is dynamically changing. To address this issue, future work will consider learning-based approaches that adapt to uncertain environments and human behavior. Reinforcement learning (RL) is a suitable tool to determine the optimal policy for uncertain systems. However, there are two technical challenges to apply RL in real environments. First, RL involves trial and errors, which can lead to catastrophic failures. Therefore, RL requires stabilizing controllers to avoid such failures. Second, the majority of RL algorithms rely on Markov assumption, i.e., full state observation. However, complete state observation is rarely available in real applications. Model-based RL approaches that run system identification for state estimator can be used to overcome incomplete state observation as in Karkus et al. (2017) and Yoon et al. (2018). The future work will focus on extending the

online HMM estimation-based reinforcement learning (Yoon et al. 2018) to a realistic environment where the flying robots and humans actively interact.

REFERENCES

Urja Acharya, Alisha Bevins, and Brittany A. Duncan. 2017. Investigation of human-robot comfort with a small unmanned aerial vehicle compared to a ground robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'17)*. IEEE, 2758–2765.

Leonard E. Baum, Ted Petrie, George Soules, and Norman Weiss. 1970. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Annals Math. Stat.* 41, 1 (1970), 164–171.

Mathias Benedek and Christian Kaernbach. 2010. A continuous measure of phasic electrodermal activity. *J. Neurosci. Methods* 190, 1 (2010), 80–91.

Dimitri P. Bertsekas. 1999. *Nonlinear Programming* (2nd ed.). Athena Scientific, Belmont, MA.

Kevin Bollino and L. Ryan Lewis. 2008. Collision-free multi-UAV optimal path planning and cooperative control for tactical applications. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*. 7134.

John Travis Butler and Arvin Agah. 2001. Psychological effects of behavior patterns of a mobile personal robot. *Auton. Robots* 10, 2 (2001), 185–202.

Jessica R. Cauchard, Kevin Y. Zhai, James A. Landay et al. 2015. Drone & me: An exploration into natural human-drone interaction. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 361–365.

Jessica Rebecca Cauchard, Kevin Y. Zhai, Marco Spadafora, and James A. Landay. 2016. Emotion encoding in human-drone interaction. In *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 263–270.

Konstantinos Charalampous, Ioannis Kostavelis, and Antonios Gasteratos. 2017. Recent trends in social aware robot navigation: A survey. *Robot. Auton. Syst.* 93 (2017), 85–104.

Yu Fan Chen, Michael Everett, Miao Liu, and Jonathan P. How. 2017. Socially aware motion planning with deep reinforcement learning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'17)*. IEEE, 1343–1350.

Ronald Choe, Venanzio Cichella, Enric Xargay, Naira Hovakimyan, Anna C. Trujillo, and Isaac Kaminer. 2013. A trajectory-generation framework for time-critical cooperative missions. In *Proceedings of the AIAA Infotech@Aerospace Conference (I@A'13)*. AIAA, 4582.

Venanzio Cichella, Isaac Kaminer, Claire Walton, and Naira Hovakimyan. 2018. Optimal motion planning for differentially flat systems using bernstein approximation. *IEEE Control Syst. Lett.* 2, 1 (2018), 181–186.

Paul De Casteljau. 1959. Outillages méthodes calcul. *Andr e Citro en Automobiles SA, Paris*.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Stat. Soc. Ser. B (Methodol.)* (1977), 1–38.

Brittany A. Duncan and Robin R. Murphy. 2013. Comfortable approach distance with small unmanned aerial vehicles. In *Proceedings of the IEEE International Conference on Robot and Human Interactive Communication*. IEEE, 786–792.

Brittany A. Duncan and Robin R. Murphy. 2017. Effects of speed, cyclicity, and dimensionality on distancing, time, and preference in human-aerial vehicle interactions. *ACM Trans. Interact. Intell. Syst.* 7, 3 (2017), 13.

Gamal Elnagar, Mohammad A. Kazemi, and Mohsen Razzaghi. 1995. The pseudospectral legendre method for discretizing optimal control problems. *IEEE Trans. Automat. Control* 40, 10 (1995), 1793–1796.

Gerald E. Farin and Dianne Hansford. 2000. *The Essentials of CAGD*. A.K. Peters, Natick, MA.

Elmer G. Gilbert, Daniel W. Johnson, and S. Sathiya Keerthi. 1988. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE J. Robot. Automat.* 4, 2 (1988), 193–203.

Edward Twitchell Hall. 1966. *The Hidden Dimension*. Vol. 609. Doubleday, Garden City, NY.

David Hunter and Geno Nikolov. 2000. On the error term of symmetric Gauss-Lobatto quadrature formulae for analytic functions. *Math. Comput.* 69, 229 (2000), 269–282.

Michiel P. Joosse, Ronald W. Poppe, Manja Lohse, and Vanessa Evers. 2014. Cultural differences in how an engagement-seeking robot should approach a group of people. In *Proceedings of the 5th ACM International Conference on Collaboration Across Boundaries: Culture, Distance, and Technology*. ACM, 121–130.

Peter Karkus, David Hsu, and Wee Sun Lee. 2017. QMDP-Net: Deep learning for planning under partial observability. In *Advances in Neural Information Processing Systems*. MIT Press, 4697–4707.

Ioannis Kostavelis, Andreas Kargakos, Dimitrios Giakoumis, and Dimitrios Tzovaras. 2017. Robot's workspace enhancement with dynamic human presence for socially aware navigation. In *Proceedings of the International Conference on Computer Vision Systems*. Springer, 279–288.

Thibault Kruse, Amit Kumar Pandey, Rachid Alami, and Alexandra Kirsch. 2013. Human-aware robot navigation: A survey. *Robot. Auton. Syst.* 61, 12 (2013), 1726–1743.

Rudolf Laban and Lisa Ullmann. 1971. The mastery of movement. ERIC. https://eric.ed.gov/?id=ED059225.

George G. Lorentz. 2012. *Bernstein Polynomials*. American Mathematical Society.

Matthias Luber, Luciano Spinello, Jens Silva, and Kai O. Arras. 2012. Socially aware robot navigation: A learning approach. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'12)*. IEEE, 902–907.

T. Marinho, A. Lakshmanan, V. Cichella, C. Widdowson, H. Cui, R. M. Jones, B. Sebastian, and C. Goudeseune. 2016. VR study of human-multicopter interaction in a residential setting. In *Proceedings of the IEEE Conference on Virtual Reality (VR'16)*. 331–331.

Michael C. Mozer, Sachiko Kinoshita, and Michael Shettel. 2007. Sequential dependencies in human behavior offer insights into cognitive control. *Integrated Models of Cognitive Systems*. 180–193. https://www.researchonline.mq.edu.au/vital/access/manager/Repository/mq:6645.

Ishaan Pakrasi, Novoneel Chakraborty, and Amy LaViers. 2018. A design methodology for abstracting character archetypes onto robotic systems. In *Proceedings of the 5th International Conference on Movement and Computing*. ACM, 24.

Panagiotis Papadakis, Patrick Rives, and Anne Spalanzani. 2014. Adaptive spacing in human-robot interactions. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'14)*. IEEE, 2627–2632.

Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. 2009. ROS: An open-source robot operating system. In *Proceedings of the ICRA Workshop on Open Source Software*, vol. 3. Kobe, Japan, 5.

Megha Sharma, Dale Hildebrandt, Gem Newman, James E. Young, and Rasit Eskicioglu. 2013. Communicating affect via flight path: Exploring use of the Laban effort system for designing affective locomotion paths. In *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 293–300.

Emrah Akin Sisbot, Luis F. Marin-Urias, Rachid Alami, and Thierry Simeon. 2007. A human aware mobile robot motion planner. *IEEE Trans. Robot.* 23, 5 (2007), 874–883.

Marshall H. Stone. 1948. The generalized Weierstrass approximation theorem. *Math. Mag.* 21, 5 (1948), 237–254.

Daniel Szafir, Bilge Mutlu, and Terrence Fong. 2014. Communication of intent in assistive free flyers. In *ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 358–365.

Daniel Szafir, Bilge Mutlu, and Terrence Fong. 2015. Communicating directionality in flying robots. In *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 19–26.

Unity Team. 2010. Unity manual. *Particle System, Positioning Game Objects*.

Benigno Uria, Marc Côté, Karol Gregor, Iain Murray, and Hugo Larochelle. 2016. Neural autoregressive distribution estimation. *J. Mach. Learn. Res.* 17, 205 (2016), 1–37.

Dizan Vasquez, Billy Okal, and Kai Arras. 2014. Inverse reinforcement learning algorithms and features for robot navigation in crowds: An experimental comparison. In *Proceedings of the IEEE-RSJ International Conference on Intelligent Robots and Systems*. 1341–1346.

Araceli Vega, Luis J. Manso, Douglas G. Macharet, Pablo Bustos, and Pedro Núñez. 2019. Socially aware robot navigation system in human-populated and interactive environments based on an adaptive spatial density function and space affordances. *Pattern Recogn. Lett.* 118 (Feb. 2019), 72–84. Retrieved from https://www.sciencedirect.com/science/article/pii/S0167865518303052.

Andreas Wächter and Lorenz T. Biegler. 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.* 106, 1 (2006), 25–57.

Larry Wasserman. 2013. *All of Statistics: A Concise Course in Statistical Inference*. Springer Science & Business Media.

Dustin J. Webb and Jur van den Berg. 2013. Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'13)*. IEEE, 5054–5061.

Felix Weninger, Florian Eyben, and Bjorn Schuller. 2014. On-line continuous-time music mood regression with deep recurrent neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'14)*. IEEE, 5412–5416.

Christopher Widdowson, Hyung-Jin Yoon, Venanzio Cichella, Frances Wang, and Naira Hovakimyan. 2017. VR environment for the study of collocated interaction between small UAVs and humans. In *Proceedings of the International Conference on Applied Human Factors and Ergonomics*. Springer, 348–355.

Hyung-Jin Yoon, Donghwan Lee, and Naira Hovakimyan. 2018. Hidden Markov model estimation-based q-learning for partially observable Markov decision process. *Arxiv preprint Arxiv:1809.06401*.

Willard I. Zangwill. 1967. Nonlinear programming via penalty functions. *Manage. Sci.* 13, 5 (1967), 344–358.

Received October 2018; revised February 2019; accepted May 2019