# Human Control Modeling

Falak Mandali, Joonwon Choi, Inseok Hwang
*School of Aeronautics and Astronautics, Purdue University, West Lafayette, USA*
fmandali@purdue.edu, ihwang@purdue.edu

***Abstract*—Data-driven models are often used in imitation learning to obtain desirable performance from autonomously controlled systems. Gaussian mixture model is one such clustering technique that is often used to train the model using data without prior knowledge. This modeling technique is successfully able to learn the characteristics of the data. However, it is not able to distinguish between desirable and undesirable behavior. This paper utilized a weighted expectation-maximization algorithm to train the GMM, and show that weighing samples of the data set allows us to control the influence of the data set on the model behavior. This is displayed using a data set from a car driving scenario, wherein the vehicle driving performance model is represented by a GMM. This work shows the implementation of environmental constraints in the model, but one could use this to incorporate human behavior constraints.**

## I. Introduction

Gaussian mixture model (GMM) is an unsupervised training model that implements clustering to determine which cluster a data point belongs to. It does not need to know to which subpopulation a data point belongs to, it learns automatically. This is accomplished by categorizing the data into subsets and creating a Gaussian model about each subset. This is performed by maximizing the likelihood of a data point of exisiting in each cluster. One of the methods that can accomplish this is Expectation-Maximizing (EM). The model then determines the probability of a datapoint to lie in each cluster/Gaussian, allowing a datapoint to exist in multiple clusters/Gaussians, otherwise known as the soft clustering technique. The adaptability of the GMM makes it useful in machine learning of data sets from various applications.

One of the most widely used applications of GMM is in the learning of robots to enable them to perform actions physically. [1] particularly focuses on the learning, planning, and optimal control of service robots that are designed to assist humans with personal or professional tasks. Due to the expectation from service robots to perform an extensive variety of tasks in an unconstrained or variable environment, the adaptability of GMMs to any application and parameter estimation technique is useful. Another popular industrial application of data-driven modeling is the safety analysis of system. The authors of [2] propose a state prediction model of human behavior to predict the response of any system. They show this by creating a GMM model using the time-indexed control inputs of the human operator controlling a drone and using Gaussian Mixture Regression (GMR) to obtain the probability distribution function of the human control inputs and the corresponding system behavior.

The above papers show that as an unsupervised learning model, GMM learns the characteristics of the data fairly well. However, this also means that the GMM does not understand what an acceptable value of a parameter is. The parameters could have a limitation on the values they can take due to either physical limitations of the system such as mechanical limitations, or due to environmental constraints. Several methods have been developed to constrain a GMM model. One set of solutions suggests not using EM for parameter estimation, and instead using gradient project [3] or sequential quadratic programming [4], amongst others. The other set of solutions recommends modifying the EM step. The authors of [5] propose implementing the P-step after the conventional EM, The P-step allows the original EM proposed Gaussians on a linearized constraint function. This method only requires the first-order derivative of the objective function as compared to the second-order that a lot of previous solutions mentioned by the [5] in their article.

Another approach to incorporating constraints is weighing each sample of a data set. The weight represents whether or not the trajectory follows set constraints. This method allows one to not discard an entire trajectory, but prevent certain sections of a trajectory from skewing the GMM. Therefore, one can still train a model with a diverse pool of trajectories or estimate parameters within limits. One can also apply this methodology when the model did have prior knowledge of the data set but somehow the data was corrupted or incorrect. One could weigh such corrupt or incorrect samples to compensate for errors in modeling. The authors of [6] prove this methodology. They derive an accurate way of incorporating weights into GMM models and choose to use EM for optimization in order to speed up computation time. The weighted algorithm is backward compatible with unweighted models, allowing for ease of application.

The objective of this paper is to create a model that imitates a human driving performance by learning from data obtained from a human driving simulator. The imitation learning model is created as a weighted GMM-EM model to account for variable trajectories driven by humans and distinguishing the valuable samples by weighing them higher, as compared to manually labeling them. This allows for extreme human behavior to be considered and estimate the driving performance within desired constraints.

This paper is organized as follows: II compares the derivation of weighted to unweighted EM, and discusses state prediction using the GMM; this is then followed by III that describes

the experiment that the data obtained from and trained for, and compares the results of the two models: conventional and weighted GMM-EM.

## II. Vehicle Driving Performance Model

### A. Gaussian Mixture Model

This work implements a modified version of the the code described in [1] to create the GMM model of the vehicle driving data. A GMM model is represented by Eqn. (1), where $\mu_i$ is the mean, $C_i$ is the covariance of each Gaussian, and $w_i$ is the ratio of the Gaussians in the GMM, and $M$ is the number of Gaussians [2]. The authors [1] propose the conventional EM algorithm to find optimal Gaussians that would describe the respective sections of data. This work modifies the conventional EM with a Weighted EM as proposed by the authors in [6].

$$P(x) = \sum_{i=1}^{M} w_i \mathcal{N}(\mu_i, C_i) \tag{1}$$

Let $L$ be a data set that the GMM trains with. Let $s_i$ be each sample point of the data set, and $\alpha_i$ be the weight corresponding to the sample point. For the weighted EM formulated below, a single weight $\alpha_i$ is applied to all elements of a multi-parameter sample point, i.e. $s_i : n \times 1$ matrix. It can seen from (6)-(9), that the weights corresponding to each sample point affect the ratio of the Gaussians, the mean, and covariance of each Gaussian, thereby changing the model.

A conventional EM is formulated as follows:

$$\text{E-step: } \eta_{i,m}^{(r+1)} = \frac{w_m^{(r)} \mathcal{N}(s_i - \mu_m^{(r)}, C_m^{(r)})}{\sum_{\tilde{m}=1}^{M} w_{\tilde{m}}^{(r)} \mathcal{N}(s_i - \mu_{\tilde{m}}^{(r)}, C_{\tilde{m}}^{(r)})} \tag{2}$$

$$\text{M-step: } w_m^{(r+1)} = \frac{\sum_{i=1}^{L} \eta_{i,m}^{(r+1)}}{L} \tag{3}$$

$$\mu_m^{(r+1)} = \frac{\sum_{i=1}^{L} \eta_{i,m}^{(r+1)} s_i}{\sum_{\tilde{i}=1}^{L} \eta_{\tilde{i},m}^{(r+1)}} \tag{4}$$

$$C_m^{(r+1)} = \frac{\sum_{i=1}^{L} \eta_{i,m}^{(r+1)} (s_i - \mu_m^{(r+1)})(s_i - \mu_m^{(r+1)})^T}{\sum_{\tilde{i}=1}^{L} \eta_{\tilde{i},m}^{(r+1)}} \tag{5}$$

A weighted EM is described as follows:

$$\text{E-step: } \eta_{i,m}^{(r+1)} = \frac{w_m^{(r)} \mathcal{N}(s_i - \mu_m^{(r)}, C_m^{(r)})}{\sum_{\tilde{m}=1}^{M} w_{\tilde{m}}^{(r)} \mathcal{N}(s_i - \mu_{\tilde{m}}^{(r)}, C_{\tilde{m}}^{(r)})} \tag{6}$$

$$\text{M-step: } w_m^{(r+1)} = \frac{\sum_{i=1}^{L} \eta_{i,m}^{(r+1)} \alpha_i}{\sum_{\tilde{i}=1}^{L} \alpha_{\tilde{i}}} \tag{7}$$

$$\mu_m^{(r+1)} = \frac{\sum_{i=1}^{L} \eta_{i,m}^{(r+1)} \alpha_i s_i}{\sum_{\tilde{i}=1}^{L} \eta_{\tilde{i},m}^{(r+1)} \alpha_{\tilde{i}}} \tag{8}$$

$$C_m^{(r+1)} = \frac{\sum_{i=1}^{L} \eta_{i,m}^{(r+1)} \alpha_i (s_i - \mu_m^{(r+1)})(s_i - \mu_m^{(r+1)})^T}{\sum_{\tilde{i}=1}^{L} \eta_{\tilde{i},m}^{(r+1)} \alpha_{\tilde{i}}} \tag{9}$$

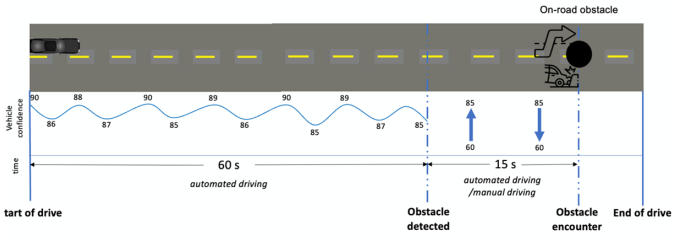


Figure 1. Experiment Scenario

### B. Gaussian Mixture Regression

A GMM represents the probability density function of the data set in the form of multiple Gaussian models. One can use Gaussian Mixer Regression (GMR) to calculate the conditional probability of the parameters of the model [2], and is described by Eqn. (10) - (13). This paper utilizes the GMR code written by the authors of [1]. It enables one to estimate the next data point given the current point and the probability density function.

$$P(x|t_k) = P(x_k) = \sum_{i=1}^{M} \hat{w}_i(t_k) \mathcal{N}(\mu_i(\hat{t}_k), \hat{C}_i) \tag{10}$$

$$\text{where, } \mu_i(\hat{t}_k) = \mu_i^u + C_i^{ut} C_i^{t^{-1}} (t_k - \mu_i^t) \tag{11}$$

$$\hat{C}_i = C_i^u - C_i^{ut} C_i^{t^{(-1)}} C_i^{tu} \tag{12}$$

$$\hat{w}_i(t_k) = \frac{w_i \mathcal{N}(t_k | \mu_i^t, C_i^t)}{\sum_j^M w_j \mathcal{N}(t_k | \mu_j^t, C_j^t)} \tag{13}$$

## III. Results and Analysis

### A. Dataset Overview

The data used to create and train a Gaussian mixture model was obtained from an experiment conducted by Myeongkyu Lee from the Graduate School of Industrial Engineering at Purdue University. The experiment was conducted to analyze the trust and bias of a human driver on how their decisions are affected by how confident the self-driving mode of the machine is in completing a task. The objective was tested by creating the following scenario: a human driver is sitting inside a car that is driving automatically at a constant speed of 55 miles per hour along a straight road. About 60 seconds later an alert is issued to the driver that an obstacle has been detected ahead. At this point, the automated driving simulator displays how confident it feels about avoiding/going around the obstacle. Knowing this information, the driver can either choose to drive the vehicle himself or let the self-driving mode continue. Refer to Fig. 1 for a visualization of the setup. It is assumed that the obtained dataset is unbiased and uncorrupted.

### B. Problem Setup

The objective of this work was to train a GMM using the driving performance data. The data set contains the performance of 20 participants of varying ages, for 12 trials each. Each trial was sampled at 60 Hz, with an average length of 75 seconds. The length of each trajectory is different as

each driver performs differently from the time of alert to the time that they have crossed the obstacle. Fig.1 shows that an average either the driver or automated driving mode will take 15 seconds to overtake the obstacles. To keep the matrices smaller for faster computation, and to be able to train more trajectories, only the driving performance data after the alert was used to train the GMM. The objective is to ensure that the vehicle does not drive beyond a certain soft constraint, i.e., the lane dimensions.

The kinematic model in [7] was used as a reference to determine parameters to train the GMM with that would best represent a car. The model requires the following data: position along the x-axis $(x)$, position along the y-axis $(y)$, longitudinal velocity of the vehicle $(v)$, heading angle $(\delta)$, and orientation angle $(\theta)$. Viewing the simulator made by the team to conduct the experiments, and using the user manual for the National Advanced Driving Simulator (NADS) - miniSim simulator [8], it was observed that the position data for the center of gravity of the car was measured with respect to an unknown reference point, and data corresponding to the orientation of the vehicle did not appear to be as the path expected from Fig. 1. In order to train the model, the position data was normalized with respect to the first data point, and the orientation angle was not used.

Two tests were run, each with five Gaussians and 15 randomly selected trajectories out of a pool of 100. This combination allowed for the maximum matrix size MATLAB could compute. Processed data was obtained from the lab team, and was organized further to be prepared for GMM training.

### C. Unconstrained GMM

This section discusses the state prediction by a GMM when no environmental or driver constraints are imposed on the GMM. The GMM is created with the model defined in (14) - (16). There is a circular obstacle of radius 5 inches located at 1160 feet from the origin, but the model does not have prior information. Furthermore, a lane boundary has been defined beyond which it is unsafe for the car to drive. The lane boundaries and obstacles can be seen in Figs. 2.

$$x_{GMM} := \{t, \delta, v, x, y\} \tag{14}$$

$$N_{Gaussians} = 5 \tag{15}$$

$$N_{samples} = 15 \tag{16}$$

Looking at the randomly selected trajectories chosen to train the model in Fig. 2, some trajectories are successful at avoiding the obstacles, whereas some either have crashed or gone off course. One can also observe erratic behavior in the vehicle longitudinal velocity and heading angle from Fig. 3 - 4. The unconstrained GMM model comprised of five Gaussians and represented with red patches in Fig.2 has a lot of deviation from the average value due to the undesirable data skewing the model. The state prediction plot further shows that the model may predict a state that could go beyond the lane limits, and/or go through the obstacle. It should be noted that the simulator data for when the vehicle crashes into the obstacle or when it steers off-course may not modeled accurately and may be unreliable.

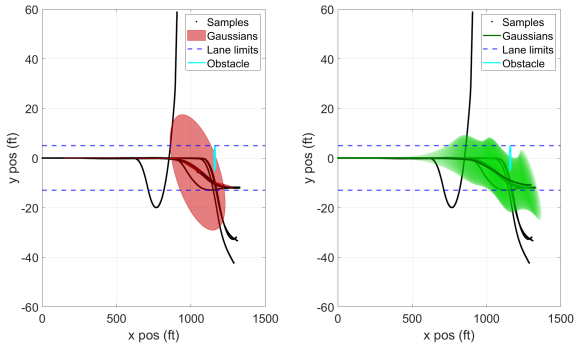


Figure 2. Unconstrained GMM: The black markers are the sample points of all the trajectories that influenced the GMM model. On the figure on the left, red patches are the five Gaussians comprising the GMM and the. It can be observed that the learned model violates lane and obstacle constraints as it does not have to distinguish between successful and failed sample trajectories. The figure on the right shows the predicted state w.r.t time. the green line represents the average of the predicted state, and the light green patch around is the first standard deviation of the predicted state w.r.t time. It shows that the predicted state also violates lane and obstacle constraints.

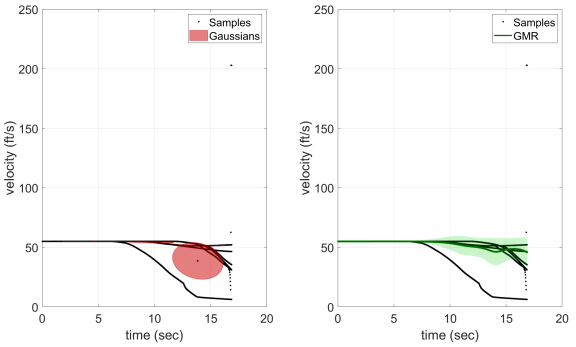


Figure 3. Unconstrained GMM: The velocity of the vehicle seems to have fairly similar behavior across trajectories, along with a few outliers. The predicted state trajectory has less variance as compared to the location of the vehicle.

### D. Constrained GMM

In order to implement a weighted EM, two constraints were applied and weighed for: vehicle never goes through the obstacle and is always within lane limits. The constraints are also defined in Eqn. (17) - (18), where $t$ is the time corresponding to when the sample is taken, $N_{Gaussians}$ is the number of Gaussians creating the GMM, and $N_{samples}$ is the number of sample trajectories used to train the GMM. If a trajectory violates either of those constraints, that section of the trajectory gets a weight of 0.01, whereas a valid section is waited as 1. By weighing each sample of a trajectory, the sections that satisfy the requirements are considered equally important for training the model when compared to trajectories that are successful throughout. This helps

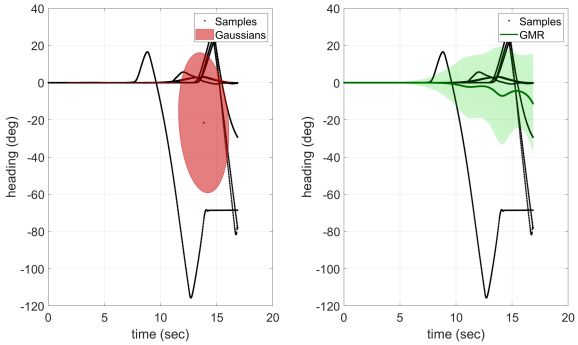


Figure 4. Unconstrained GMM: The vehicle heading data seems to be changing very erratically for some cases that are skewed by the trained model. The predicted state trajectory proposes a wide range of possible states that may not be feasible in real life.

in preventing the GMM from being too conservative and allowing for wider variation in states. This can be observed in Figs. 5, wherein the sections of the trajectories in black have a weight of 1, and tend to influence the GMM more than the trajectories in gray that have a weight of 0.01.

It can be observed from Figs. 5 that the weighted GMM is not skewed by undesirable trajectories. One of the Gaussians appears to be crossing through the obstacle, but the predicted state neither violates lane boundaries nor goes through the obstacles. The obstacle avoidance limit was set to be the radius of the obstacle, adding a small safety region around the obstacle would result in a slightly better model. Similarly, heading angle Fig. 7 GMM and state prediction appear to have less violent behavior than the unconstrained case. The black and gray color scheme corresponds to the vehicle's performance within the constraints mentioned in Eqn. (14) - (16). The behavior of the velocity of the vehicle shown in Fig. 6 is similar to the unconstrained case shown in Fig. 3.

$$||p_{vehicle}(t) - p_{obs}|| > r_{obs} \tag{17}$$

$$p_{vehicle} < p_{lane} \tag{18}$$

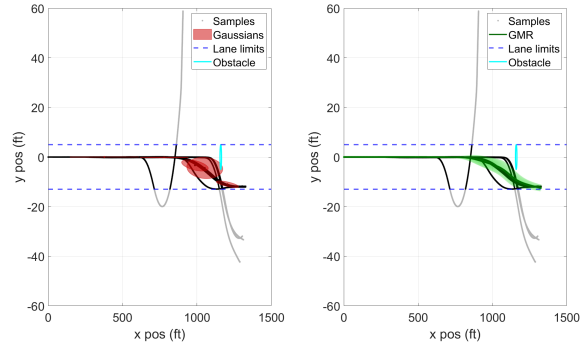


Figure 5. Constrained GMM: The low weights are only applied to sample points of the trajectories that are outside the constraints. The GMM provided a safe prediction of the state that does not violate either boundary constraint.

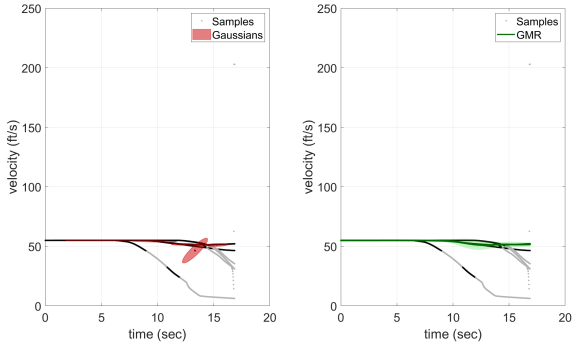


Figure 6. Constrained GMM The vehicle velocity model appears to be similar to the unconstrained case as there is not a lot of variation in the data.

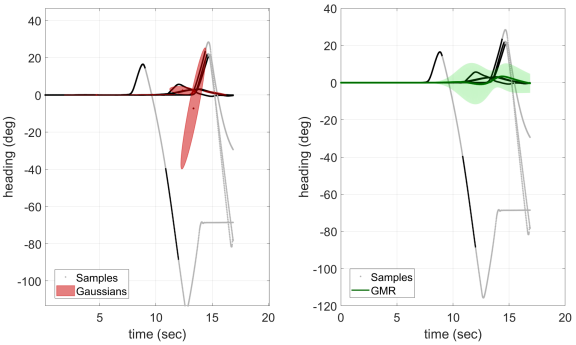


Figure 7. Constrained GMM: The GMM and state prediction horizon is much smaller than the unconstrained case, and appear to be more feasible in real-life application.

## IV. Conclusion

This paper aims to create a data-driven model that imitates human vehicle driving performance. A Gaussian mixture model (GMM) was trained to represent the vehicle driving performance with respect to time. Such a model could be implemented in an autonomous vehicle to make vehicle movement safer. A weighted expectation-maximization algorithm was used to specify environmental constraints. The results of the weighted GMM were compared against the unconstrained GMM and it shows that the weighted model allows for the exclusion of undesirable sections of sample trajectories and prevents those sections from skewing the GMM. This enables the user to train a GMM model with constraints, which can be added during data processing, and train a data set without manually having to filter out undesirable samples. This type of modeling can be used not only include environmental constraints, but also human behavior constraints that could further improve other vehicle performance matrices such as velocity, acceleration, turns, and more.

## References

[1] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," vol. 9, no. 1, pp. 1–29. [Online]. Available: http://link.springer.com/10.1007/s11370-015-0187-9

[2] J. Choi, S. Byeon, and I. Hwang, "State prediction of human-in-the-loop multi-rotor system with stochastic human behavior model," vol. 55, no. 41, pp. 119–124. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2405896323001209

[3] M. Jamshidian, "On algorithms for restricted maximum likelihood estimation," vol. 45, no. 2, pp. 137–157. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0167947302003456
[4] B. Li, W. Liu, and L. Dou, "Unsupervised learning of gaussian mixture model with application to image segmentation," vol. 19, no. 3, pp. 451–456. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10167432
[5] K. Takai, "Constrained EM algorithm with projection method," vol. 27, no. 4, pp. 701–714. [Online]. Available: http://link.springer.com/10.1007/s00180-011-0285-x
[6] D. Frisch and U. D. Hanebeck, "Gaussian mixture estimation from weighted samples," in *2021 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pp. 1–5.
[7] R. Pepy, A. Lambert, and H. Mounier, "Path planning using a dynamic vehicle model," in *2006 2nd International Conference on Information & Communication Technologies*, vol. 1, pp. 781–786.
[8] miniSim™. [Online]. Available: https://www.nads-sc.uiowa.edu/minisim/