Marilena Lemonari\(^1\), Andreas Panayiotou\(^1\), Nuria Pelechano\(^1\), Theodoros Kyriakou\(^1\), Yiorgos Chrysanthou\(^1\), Andreas Aristidou\(^1\), and Panayiotis Charalambous\(^1\)

\(^1\)Affiliation not available

January 29, 2024
P2C: A Paths-to-Crowds Framework to Parameterize Behaviors

Marilena Lemonari*, Andreas Panayiotou*, Nuria Pelechano, Theodoros Kyriakou, Yiorgos Chrysanthou, Andreas Aristidou, Senior Member, IEEE, and Panayiotis Charalambous

Abstract—Simulating believable crowds heavily relies upon the perceived realism and diversity of the agents’ behaviors, whilst generating novel crowd simulations highly depends on being able to easily manipulate intuitive behavior parameters. We present P2C (Paths-to-Crowds), a method that parameterizes reference crowd data and enables the simulation of similar behaviors in different environments. This approach enables explainable and fine-grained control of simulations since artists can modify at run-time a small set of intuitive parameters in local regions. We incorporate the integration of four fundamental behaviors: goal-seeking, grouping, interaction with areas of interest, and connectivity-into an existing Reinforcement Learning (RL)-based crowd simulation system, which facilitates the creation of customizable agent behaviors and interactions. To learn a parameter model, we synthesize numerous simulations by sampling the parameter space; we then use these data to learn a model that outputs the underlying parameters. To achieve this, we encode the simulation in a set of 2D maps that encode different measurements such as velocities, occupancy, interpersonal distances, path deviations, etc. The trained model can then be used to infer parameters in localized regions of given crowd data (both real and simulated). This approach enables replication of behavior, transfer to new environments, real-time local control, editing of parameters, and explainability of behaviors (with respect to the fundamental behaviors). We evaluate our model’s predictive power on real data and compare it against existing baselines. This along with the accompanying user study reveals P2C’s potential in terms of achieving behavior realism and diversity.

Index Terms—Neural networks, Interactive simulation, Crowd simulation, User control, Crowd authoring, Data-driven methods, Interactive tool

1 INTRODUCTION

Crowd simulation is an important topic for research having an immense impact on various industries such as film and gaming, training and evacuation systems, urban planning, and architectural visualizations. Generating realistic and diverse behaviors for virtual crowds is a crucial attribute for a successful simulator. In the past, agents were simulated to perform basic tasks such as reaching a designated point without colliding with their surroundings [49]. As the domain continues to advance, the primary objective is diversifying crowds by integrating more sophisticated behaviors that mirror those observed in real crowds. This necessitates the development of more complex interactions between agents and their environments. Previous research has mainly focused on investigating grouping behaviors, formations, and the interactions of agents with specific areas of interest within a scene [25], [45]. However, real people frequently demonstrate intelligent behavior by simultaneously exhibiting multiple actions. Creating plausible simulations requires not only modeling individual behaviors but also accurately representing the intricate interplay between these behaviors within the environmental context. For instance, pedestrians may progress towards a goal while maintaining proximity to their “friends” and adjusting their pace when something in the environment captures their attention. The challenge of creating realistic crowd behaviors lies in understanding the complex behavioral patterns and contextual relationships, including environmental cues, emphasizing the significance of integrating knowledge to achieve authentic and convincing crowd simulations.

In addition to automatically generating plausible trajectories that accurately capture real behaviors, it is equally significant
to provide users with an interface that offers creative control in an easy and meaningful manner. The effectiveness of such a user interface (UI) is directly linked to how easily users can comprehend the impact of manipulating simulation parameters on the outcomes. While there are several authoring interfaces customized to enhance specific functionalities, such as drawing UIs for managing crowd flow [6], finding UIs capable of concurrently controlling multiple simulation components is challenging. This underscores the importance for designing and developing user-friendly interfaces that seamlessly integrate the capabilities of complex simulation frameworks, thus further facilitating the generation process. This design will enhance usability not only for novice users but also for industry professionals, expanding its potential for widespread use.

Undoubtedly, improving the perceived realism and plausibility of behaviors in simulated crowds is a crucial first step in developing an effective synthesis tool. The use of real behaviors extracted from example data, such as videos, can greatly facilitate the generation of realistic results, making them suitable for replicating observed behaviors and allowing for the synthesis of new crowds by manipulating and blending existing behaviors. However, these methods are constrained by the input data, as behaviors are sometimes treated as black-box information without parameterization, limiting the creative freedom of authors [30]. On the other hand, simulated crowds, though not perfectly replicating real data, can still appear believable and offer greater flexibility in crowd synthesis. Nevertheless, their limitations lie in how simulator parameters are defined. In terms of authoring, users often face a trade-off between ease-of-use and the level of realism. More intuitive tools, for example, provide macroscopic control, but achieving individuality often comes with tedious manual parameter tuning.

In this research, we develop a mapping from crowd data to intuitive crowd parameters, enabling behavior parameterization from in-the-wild videos. Initially, we build a synthetic dataset (see Section 3.2) containing encoded images and their corresponding ground truth behavior weights. This dataset is generated by sampling the crowd parameter space, as defined in Section 3.1. Subsequently, using this dataset, we train a convolutional network (see Section 3.3) capable of taking images encoded from any set of trajectories and producing sets of explainable behavior weights. It is essential to note that our framework currently addresses only four fundamental crowd behaviors, and while it does not encompass all possible crowd behaviors, its impact lies in its ability to parameterize crowd data within the selected parameter space. To illustrate the integration of these fundamental behaviors, which include goal-seeking, grouping, object interaction, and connectivity, we utilize the Configurable Crowd Profiles (CCP) model [37]. CCP is selected as our underlying parameter model due to its ability to also capture mixtures of behaviors, but it is important to note that our method is adaptable and can be applied to other models as well. Our framework, Paths-to-Crowds (P2C), enables the generation of novel crowds (see Section 3.4), and through an authoring tool (see Section 3.5) allows users to create a custom environment and synthesize crowds exhibiting behavioral patterns implicitly defined by a given reference video. During inference, the model identifies optimal simulation parameters for unseen trajectories from unconstrained input videos, and then users can edit and distribute the output parameters across a customizable new environment using our UI. We assess our model’s predictive power on real data and compare it with existing baselines. Our experiments reveal that P2C can generate behaviors statistically close to real ones, consistently achieving higher behavior similarity than other simulators such as RVO [48] and CCP “baselines”. Additionally, we conduct a user study for real/P2C/baseline comparisons, indicating a high plausibility for P2C simulations, despite various factors influencing the perceived realism of behaviors in virtual crowds.

In summary, our primary contributions include:

- A method for learning simulator parameters, providing precise control and generation of explainable outputs.
- Generalization of output parameters, allowing for the transfer to custom environments without compromising simulation plausibility.
- Capturing of parameters that exhibit variations in both space and time.
- A simulated training dataset consisting of pairs of image encodings and corresponding behavior parameters.

Finally, we plan to make our code and data accessible online to allow others to expand on our work and results.

2 RELATED WORK

Several aspects make a virtual crowd believable, mainly behavior realism and diversity. Thus, we discuss past works on diversifying crowds and on using data-driven methods to improve realism. We review past literature on methods basing their inputs on videos, identifying the gaps, and stating where our work fits in the field. Finally, we present the current state of the crowd authoring domain, allowing behavior control at several stages of the simulation.

2.1 Behavioral Diversity in Crowds

Introducing heterogeneity in crowd movements surely gives a realistic feel of the outcome since real people also act and move in a variety of ways even within groups/crowds. Several ways have been used in research to replicate this effect. Essentially, it comes down to defining a set of parameters and then developing methods that give appropriate values to these parameters. Early works use simpler predefined weights [42], [44] like acceleration, while more recently, researchers incorporate the idea of personality traits to drive their parameter definition [13], [24]. For example, Durupinar et al. [8] incorporate “personality” by associating a set of low-level parameters with the five OCEAN model personality traits. This automates low-level parameter specification since the OCEAN parameters are intuitive and easy to associate with the effect they have on the simulation. For example, when increasing “openness”, it is expected that agents will prolong their stay in the environment. Also, Ren et al. [41] tackle grouping weights via a unified velocity approach, achieving multiple group dynamics abiding by user constraints. Even so, the results of these frameworks are limited to how they define their behavior parameters. Panayiotou et al. [37] utilize an RL-based method to learn multiple behaviors simultaneously, hence allowing for the emergence of diverse crowds not only with distinct behaviors, but also with a mixture of dominant behaviors like goal-seeking, grouping, and environment interactions. Still, for users to author a new crowd behavior, they need to experiment with these intermediate behavioral states that often have small, subtle differences. Instead, in our framework, the users only need to give as input an example video (and trajectories), which shows a real crowd with the desired
overall behavior, and then the specific behavior weights are given directly, allowing them to distribute those in new environments according to their preferences. Similarly to our work, Charalam-bous et al. [5] utilize novelty detection on trajectory data to define an imitation reward during training of individual agents in an RL setting. This method achieves diverse behaviors seen in the source data, however, it works on the individual agent level, policies are trained on different datasets and does not allow for easy editing of behaviors.

### 2.2 Data-Driven Crowds

Despite the advanced state of the literature on crowd heterogeneity, there is much potential in learning essential parameter values from data. Data-driven methods expanded the crowd simulator capabilities, gaining popularity in the early 2000s and still being highly relevant to this day. In particular, there are two main reasonings for data-driven crowd simulation: (1) trying to imitate behavioral patterns of some reference data, or (2) using data to learn a set of parameter values.

For the former, methods were initially graph-based and used to achieve flocking and navigation behaviors in general [26], [27], [52], not particularly addressing behaviors diversity. Crowd Patches [52] achieve high scalability but do not allow for dynamic events or real-time interactivity, and require manual work to add variety to the pre-computed paths, while our method handles this automatically. A more advanced graph-based method by Charalambous and Chrysanthou [3] encodes the agents’ states using temporal representations and construct a Perception-Action Graph (PAG) by grouping similar Temporal Perception Patterns (TPPs) in the graph nodes. Pioneer work by Metoyer and Hodgins [34] integrates an interactive component to allow users to give the desired example behaviors for the crowds. Later on, researchers attempted to replicate the heterogeneity observed in real crowds, via letting agents learn by a set of examples [28], [30], [53]; additional subtle behaviors increase the perceived realism [32]. Another way crowd diversity is achieved is by blending the captured behaviors from the input data [19]. More recently, Peng et al. [39] use imitation learning to guide an RL system so that it can follow examples from videos. However, these methods are limited by the input data and hence fail to exhibit generalizability qualities.

On the other hand, there is an opportunity to optimize crowd parameters using real data. Most of literature based on this, builds some method to analyze the real data in a way such that the analysis outcome has compatible structure with the parameters to be optimized. Often-times, navigation and avoidance parameters are studied e.g., time-to-collision, since early works focused on common simulator parameters [38], [40], [47]. Regarding more high-level parameters, several works generate simulated data which they compare to the reference real data [4], [14], [16], [22], [23], [50], [51].

Also, frameworks that have videos as input approach this problem either as a computer vision task which outputs videos, or as a simulation task that uses video information to synthesize crowds in the 3D space. An example of the former by Flagg and Rehg [11] designs crowd tubes to place segmented pedestrians in a new video constraint based on environmental characteristics. We deal with the latter, since the vision-based results are limited to frame prediction [2], and lag behind in capturing realistic behaviors.

The surge in popularity of ML algorithms has reshaped approaches to optimization tasks. Notably, Panayiotou et al. [37] employ Reinforcement Learning (RL) to optimize model parameters, achieving a mixture of behavior weights that accurately represent complex human behaviors, yielding the CCP parameters. In contrast, our approach focuses on achieving a mapping that enables the parameterization of any behavior, as we do not impose restrictions on the input videos.

Morphable crowds by Ju et al. [19] employ a formation and a trajectory model managing to blend real input data to synthesize crowds with interpolated behaviors. Our work differs, in that we do not manipulate the real trajectories but instead, we generate parameters leading to simulated trajectories that perform the behaviors of the input. We emphasize that our framework extracts behaviors from in-the-wild videos; no need to have description of the behaviors captured in the videos since we map them to easily interpretable parameters. Other relevant works sample trajectories to find the state of each agent, whereas we study collective behaviours per area [19], [37]. Our approach is mesoscopic (neither microscopic nor macroscopic) as we observe the state of an area rather than individual agents.

Perhaps closer to our concept is the work of Lee et al. [28], where an agent model is learned from video-extracted trajectories. This model generates agent actions, considering the environment and neighboring movements, resulting in versatile crowds. Despite the remarkable capabilities of their system, our approach differs in that each generated crowd profile comprises a blend of multiple intricate behaviors. This allows us to employ input videos that may not readily accommodate straightforward and controlled behavior annotations. Consequently, when users author new environments, they can utilize these predicted advanced behaviors as prefabs, sparing them the tedium of detailed descriptions. Our emphasis lies in the aspiration to ultimately facilitate novel simulations in customized environments, incorporating new realistic behaviors inspired by real-world dynamics rather than solely replicating behaviors captured in videos.

### 2.3 Authoring Crowd Behaviors

User intervention is essential in realizing the authors’ creative vision. Therefore tools for authoring virtual crowds are crucial components in most simulation frameworks. Crowd simulation is a complex field making authoring a broad and multi-level task, as the user can interfere in several stages of the generation [29]. Authors can have influence over one or more crowd simulation components such as high-level behaviors, path-planning, local movements, animation, visualization, and post-processing. For each component, there are several types of tools that have been developed in past works, to allow for that level of user control. For example, drawing and sketching interfaces are popular for path specification [35], while more manual parameter specification is used for defining local movements [9]. High-level parameters like personality traits and agendas implicitly affect all other components [8], [43]; Albeck’s CAROSA system [1] employs Microsoft Office tools to control agent responsibilities.

In previous approaches related to our concept, crowd behavior was controlled using deformation gestures for formations and crowd flow [18], weight manipulation for grouping [41], and user-friendly tools such as brushes, storyboards, and sliders [21], [37], [46]. Another notable contribution is Interaction Fields by Colas et al. [7], which provides an intuitive authoring interface.
allowing users to draw, through simple mouse movements, behaviors centered around a source point (object or agent), creating an “interaction field”. This field influences the movements of agents within its range. However, users need to familiarize themselves with the system through trial and error, setting up the simulation from scratch. In contrast, our interface presents suggested behaviors to users that were driven by real behaviors observed in videos. Consequently, users need only to refine the results according to their intentions, using intuitive functionalities like environment augmentation buttons and slider manipulation for dominant behavior weights. Our approach encourages users to design a new environment and distribute behaviors implicitly defined from a reference video (or set of paths) as they see fit. It also facilitates the authoring of both local movements and path-planning, as well as high-level behaviors via the intuitiveness of the parameter set.

3 The P2C Pipeline
This section provides a breakdown of the P2C pipeline: the training setup (Figure 2) and inference applications (Figure 1). More specifically, we discuss (1) the dataset generation of synthetic images of trajectories with their corresponding crowd parameters, (2) the CNN-based model and the respective training framework, (3) how the model is utilized to serve the P2C framework, and (4) the authoring tool available for novel synthesis and further customization of the resulting simulation.

3.1 Underlying RL-based Behavior Model
We modified the open-source CCP framework [37], which gives the ability of capturing a variety of behaviors concurrently by blending multiple “core” behaviors; we adjust the action and observation space, while also we re-balance the reward function. Our choice to use CCP stems from the need to have the effect of multiple behaviors in a single simulation, easily with intuitive and adjustable behavior weights. Even so, we claim that our proposed methodology could be modified to support any other simulator parameters corresponding to behavior attributes.

We adjust the action space to be continuous, contrary to the original implementation which as discrete; we move agents by calculating a preferred velocity at each action step, based on the actions generated by the policy network, which include a moving distance and a rotation angle. The range of moving distance is $[-7.2] m/s$, while the range for the rotation angle is $[-45, 45]$ degrees. We also pass the calculated velocity to the RVO simulator [48] which handles the movement and ensures collision avoidance between agents and obstacles; a new preferred velocity is generated every $5$ simulator steps ($0.2 \times$ timestep).

The observation space is expanded by combining the existing ray sensor with a grid sensor that uses a set of box queries that provides a top-down 2D view around the agent; both sensors provided by Unity’s ML-agents framework [20]. This modification allows agents to get a better “sense” of the environment, and plan their future actions in a more sophisticated way.

We modify the reward function and removed penalties for collision avoidance, as we decided to use RVO to handle them. We introduce a new weight called “connectivity” ($w_c$), that controls how close to each other the agents are while executing their current behavior; we keep goal seeking, grouping, and interaction with points of interest, as our core behaviors. This addition enables the simulation of behaviors such as groups of people that are walking together, while also controlling how close together stationary groups are. In order to implement this, we first spawn agents in clusters, then we positively reward the agents if they keep the speed variance among their cluster low, and keep their distance to the cluster’s center-of-mass near the desired. Likewise, the conditions for the other three behaviors are staying similar to the original work:

- Goal-Seeking: moving towards goal position.
- Grouping: standing near other agents, looking towards the center of the group, $\# \text{ of neighbors lower than a threshold}$.
- Interaction: standing near a POI, looking towards POI, $\# \text{ of neighbors lower than a threshold}$.

Finally, besides the individual behavior rewards, we use a smoothing reward that encourages agents to avoid selecting a future velocity that is very dissimilar to their current one, thus enforcing smooth movement.

Training is done by spawning agents randomly in the environment in groups of $1$ to $5$. Then, we randomize the environment (obstacles, POIs) and incorporate a curriculum-based approach as described in the original work.

3.2 Dataset
Next, we create a dataset with (image, profile parameters) pairs, to be used during training, with the aim of obtaining ground truth pairings that connect an image with a set of simulator parameters; we plan to release this data. For the parameter space, we choose behavior weights as described in Section 3.1 $\{w_g, w_{gr}, w_i, w_c\} \text{ i.e., weights for goal-seeking, grouping, interaction and connectivity}$ respectively, all within the range of $[0, 1]$ with the additional constraint that $w_g + w_{gr} + w_i + w_c = 1$. For the purposes of this research,
each set of behavior weights is considered a crowd profile; it uniquely identifies the desired behavior of a crowd or a set of agents. For a specified crowd profile, we randomly generate the environment setup i.e., obstacles and POIs, initialize agents, and let them move for \( \sim 20s \) (enough time to exhibit their assigned behavior) documenting their trajectories. For the image space, we use the documented trajectories to encode important information in five-channel images; if our data came from a structured setting, we could have alternate representations, however we argue that images is a suitable option for unstructured data such as information from in-the-wild crowd videos. Likewise, since we aim to predict behavior parameters, we find it necessary to incorporate spatial information in the inputs. Each of the encoded input channels corresponds to:

1. the horizontal velocity,
2. the vertical velocity,
3. the most efficient path to the goal point (optimal path),
4. the interaction clusters both between agent groups and around points of interest,
5. and lastly the values of interpersonal distances between groups throughout the simulation.

Note that we refrain from using the trajectory image alone since it does not give context about the direction of movement of agents e.g., an agent moving left-to-right and right-to-left will yield to the same visual trajectory in the generated image. Thus, we encode velocity instead which helps the model understand the navigation component. The remaining channels are encoded as follows. The optimal path is constructed by drawing a straight line connecting agent’s spawn and goal positions; the value of the line’s pixel is the inverse of the combination of agent’s average path deviation and the total path distance. The interaction cluster is generated by connecting the positions of agents, in a radius of 3.6m (social distance as defined by Hall [15]), that are stationary in similar time intervals; the intensity of the line is based on total duration that two agents are “connected”. We also visualize the POIs in this channel, if any in the scene. Finally, the interpersonal distance visualizes the center-of-mass trajectory of each cluster; agents are assigned to clusters based on distance (≤ 2.5m) and the overall trajectory similarity; the pixel’s value represents the cluster’s average interpersonal distance at the current time.

We build this dataset by randomly generating weight values which is mapped to the image space via generating and stacking the five channels, thus producing the respective (5, 64, 64) images; Figure 3 shows an example of these image encodings from a reference trajectory image. Since the images are intended to be used as inputs to the resulting model, we encode additional information via the pixel intensity as mentioned before. Firstly, for the Velocity channels, pixel intensity corresponds to positive or negative speeds. Secondly, for the Optimal Path channel, lighter pixels represent lower path deviation and overall path distance. Thirdly, for the Interaction Cluster channel, lighter pixel values represent longer grouping duration between agents. Finally, for the Interpersonal Distance channel, lighter pixels implies closer proximities between agents in the same group. The simulations were done in an area of 14x14m, which is an appropriately sized area for the spawned agents to execute their behaviors. For this areas, we spawn between 4 and 10 agents; we found that this enables the agents to exhibit their assigned behaviors, while also maintaining similar scales compared to the real world data. The resulting synthetic dataset contains 150K (image, profile parameters) pairs; we use 75% of that for training, 15% for validation and 10% for testing.

An ablation study on the input channels has been conducted to access the contribution of the aforementioned encodings, by gradually removing individual channels from the inputs of the CNN-based model and documenting the model losses and accuracies in Table 1. Note that we did not remove the velocity channels as we consider them essential for the trajectory representations. We carry out the experiments with a smaller dataset of 25000
Accuracies

<table>
<thead>
<tr>
<th>Channels</th>
<th>Train</th>
<th>Val.</th>
<th>(w_{\text{goal}})</th>
<th>(w_{\text{group}})</th>
<th>(w_{\text{inter}})</th>
<th>(w_{\text{conn}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>All (5)</td>
<td>1.019</td>
<td>1.613</td>
<td>0.786</td>
<td>0.669</td>
<td>0.733</td>
<td>0.763</td>
</tr>
<tr>
<td>w/o OP</td>
<td>1.031</td>
<td>1.620</td>
<td>0.756</td>
<td>0.642</td>
<td>0.733</td>
<td>0.762</td>
</tr>
<tr>
<td>w/o IC</td>
<td>1.150</td>
<td>1.708</td>
<td>0.731</td>
<td>0.559</td>
<td>0.568</td>
<td>0.749</td>
</tr>
<tr>
<td>w/o ID</td>
<td>1.018</td>
<td>1.602</td>
<td>0.797</td>
<td>0.682</td>
<td>0.747</td>
<td>0.755</td>
</tr>
<tr>
<td>only OP</td>
<td>1.150</td>
<td>1.703</td>
<td>0.735</td>
<td>0.543</td>
<td>0.553</td>
<td>0.612</td>
</tr>
<tr>
<td>only IC</td>
<td>1.035</td>
<td>1.617</td>
<td>0.767</td>
<td>0.661</td>
<td>0.726</td>
<td>0.737</td>
</tr>
</tbody>
</table>

samples, while keeping the hyperparameters constant throughout the experiments for fair comparisons i.e., 150 epochs, batch size of 256, and learning rate of .0005.

Based on the findings of Table 1, we can conclude that the less significant channel is that of the interpersonal distances since removing it yields the lowest losses and highest accuracies, except of course the connectivity value accuracy. Still, it is seen that its differences with the full inputs model are almost negligible, therefore we choose to proceed utilizing all five encodings as input channels to our model.

For further visualization of the encodings, Figure 4 illustrates examples of trajectories and encodings for weight dominant behaviors i.e., goal-dominant, group-dominant, and interaction-dominant profiles. These examples confirm our intuition that the optimal path channels is useful in assessing how prominent the goal weight is, whereas the interaction cluster channel gives valuable information regarding the impact of the group and interaction weights.

![Fig. 4: Channel encoding along with their reference trajectory images for an example goal-dominant behavior (first row), group-dominant (second row), and interaction-dominant behavior (third row), respectively.](image)

### 3.3 Training

Having built the dataset, we proceed to train a CNN-based model to take as input a five-channel image (as described in Section 3.2) set and output the underlying crowd profile that matches the behaviors observed in the image. Our objective is to find a mapping between the image space and the crowd parameter space so that it is possible to observe behaviors in-the-wild and parameterize them in a way that is: understandable (weights of intuitive behaviors), adjustable (easy user intervention), and can be used as building blocks for novel, custom simulations (via authoring interface).

As mentioned before, we choose to use the four parameter weights as described in Section 3.1, since we believe they are capable of spanning a wide range of intermediate and sophisticated behaviors. Of course, our framework can be adjusted to support other crowd parameter spaces as well.

During training, we randomly apply transformation operations to the images i.e., rotation and flipping. As mentioned before, the input of our CNN-based model is a five-channel 64x64 image and the final output corresponds to the four behavior weight values. The model has three convolutions (kernel size: 3) and four fully-connected layers. The output of the last linear layer is a six-valued vector. The full model architecture is shown by Figure 5.

![Fig. 5: Model Architecture: A five-channel 64x64 image is given as model’s input and after the convolutional layers, a set of fully-connected layers is applied ending with 6 neurons. The first three are used to rank the goal, group, and interaction behaviors according to dominance. The next two predict the differences between the most dominant and second most-dominant, and the most and least dominant behaviors. Having the ranked behaviors along with the behavior deviation, the three weight values are then computed, while the connectivity weight is directly given by the last output.](image)

Note that, the model by design, does not directly regress the four behavior weights. This model design yields better scores and needs less training data. We presume that the higher performance is due to the limited prediction space; by first having the ranking, the model is given further context and so limits the kind of combinations to predict. The first three values of the output are used for ranking the dominant behaviors \([b_{1st}, b_{2nd}, b_{3rd}]\), trained with a cross entropy loss. The next two values \((d_1, d_2)\) are designed to represent the differences of the most dominant and second and third most dominant behaviors respectively i.e., \(d_1 = w_{1st} - w_{2nd}\), and \(d_2 = w_{1st} - w_{3rd}\). From the outputted differences, the predicted weights \(w_g, w_{gr}, w_i\) can be calculated according to Equation 1, and are trained with L1 loss.

\[
\begin{align*}
    w_{1st} &= (1 + d_1 + d_2)/3w_{2nd} = w_{1st} - d_1 w_{3rd} = w_{1st} - d_2 \\
\end{align*}
\]

Lastly, the final outputs is set to correspond to the connectivity value \(w_c\), and also trained with the regression loss; we isolate connectivity since it can coexist with all other behaviors and serves more as a distance modifier rather than a core behavior. Hence, the loss function is given by:

\[
\begin{align*}
    loss_{total} &= 0.7 \cdot loss_{R1} + 0.3 \cdot loss_{R2} \\
    &= loss_{L1}(w_{1st}) + loss_{L1}(w_{2nd}) + loss_{L1}(w_{3rd}) + loss_{L1}(w_c) \\
\end{align*}
\]

where \(loss_{R1}\) and \(loss_{R2}\) are the entropy losses of the most dominant and second most dominant behaviors, respectively. We prioritize the regression weights by giving them high importance because they correspond to our ultimate outputs, while giving
higher importance to the first (0.7) rather than the second (0.3) most dominant behavior because the dominant is the one having the largest impact on the resulting behaviors; the specific values are chosen via trial-and-error.

After the model is trained, we use the validation set and a tolerance threshold of 0.1 to compute the individual accuracies. We choose this tolerance value since we observed that smaller differences in weight values do not have noticeable impact in the resulting behaviors. Our model is able to predict the 1st-most-dominant and 2nd-most-dominant behaviors with accuracies 87% and 74% respectively. Furthermore, regarding the individual behaviors, the model predicts the goal-seeking, grouping, interaction and connectivity weights with accuracies 79%, 68%, 75% and 77% respectively.

Prior to training the full model, we conduct an additional ablation study to emphasize the need for the four aforementioned behaviors (Table 2). We fix hyperparameter values and train smaller models, documenting the difference between the distance covered by the agents following our models and the true distance covered by ground truth agents. We compare with three snippets of real data from a busy street, a university campus, and a church yard (Zara, Students, and Church [3], [31], respectively). Our choice of using all four behaviors as on average, the full model performs best. The per-dataset results are also sensible since the Zara data mostly consist of people walking to their goals, thus removing the goal behavior hurts the performance the most. In a similar manner, the church data does not contain any interaction objects, hence removing the interaction behavior is actually beneficial for the performance. Having in mind the real world behavior diversity in different scenarios, we base our choice of using all four behaviors on the average metric of the three data clips.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Zara</th>
<th>Students</th>
<th>Church</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Full Model</td>
<td>2.908</td>
<td>4.972</td>
<td>3.258</td>
<td>3.713</td>
</tr>
<tr>
<td>w/o Goal</td>
<td>10.173</td>
<td>6.555</td>
<td>10.728</td>
<td>9.152</td>
</tr>
<tr>
<td>w/o Group</td>
<td>4.267</td>
<td>7.773</td>
<td>8.096</td>
<td>6.712</td>
</tr>
<tr>
<td>w/o Inter.</td>
<td>4.175</td>
<td>6.043</td>
<td>2.711</td>
<td>4.310</td>
</tr>
<tr>
<td>Fixed Conn.</td>
<td>2.888</td>
<td>5.761</td>
<td>5.196</td>
<td>4.615</td>
</tr>
</tbody>
</table>

The table shows the need of our chosen behaviors as on average, the full model performs best. The per-dataset results are also sensible since the Zara data mostly consist of people walking to their goals, thus removing the goal behavior hurts the performance the most. In a similar manner, the church data does not contain any interaction objects, hence removing the interaction behavior is actually beneficial for the performance. Having in mind the real world behavior diversity in different scenarios, we base our choice of using all four behaviors on the average metric of the three data clips.

### 3.4 Model Utilization

After the regression model is trained and validated, it is used on data from real video trajectories. Simply, a real-world video is discretized both in time and space as follows: we use an appropriate time window for each dataset (i.e., 125, 250 or 500 video frames), and according to the aspect ratio we divide the space into smaller areas that will each have their own behavior profile, as hinted in Figure 6.

All three videos have aspect ratio of 4 : 3, which corresponds to the size of the grid. Here, we emphasize that training was conducted with a 14x14 environment, so we discretize to bring the test data closer to the training data; with our test videos, we found it best to split it in those 12 areas. Of course, this is not universal to all videos, motivating further research as to how this discretization should happen. Given the space and time discretization, the agent trajectories are extracted specifically for the current behavior area and current timeframe, to create an input image set (5, 64, 64). Then, the image sets for each area are fed into the trained model resulting in a series of predicted profiles for all the temporal and spatial video splits.

### 3.5 Authoring Tool

As a last component in our pipeline, we allow user intervention via an authoring interfaces which facilitates (1) synthesizing new crowds that behave according to the input behaviors, and (2) manual tweaking of parameter weights. Our main authoring tool functionality enables users to populate a different virtual environment with crowds that move according to the extracted video behaviors. Figure 7 shows the overview of this authoring tool hinting at its functionalities: (1) the user creates a custom environment by creating the layout of the area, the positions and scales of obstacles/POIs, and spawn/goal positions of agents, (2) then a clustering algorithm presents to the user the suggested behaviors, as those were predicted by our model, and finally (3) the user assigns them to specific environment areas for specific timeframes.
We use the DBSCAN clustering algorithm [10], with parameters $\epsilon_{min} = 125$ and minSamples $= 1$, to group together similar predicted behaviors and show the user the cluster centers as the behavior-dominant profiles (prefabs) to further ease authoring. Apart from that, we allow users to alter the replicated behaviors to fit their wishes by enabling them to choose a specific behavior area - specifying the desirable timeframe and grid position - and adjust the set of predicted parameter values with simple sliders. We alleviate the need to manually specify the parameter values from scratch which would mean that the user should have been able to find the link between the observed behaviors and the respective parameters via trial-and-error.

4 Results

Our model has been trained using an AMD Ryzen 9 7900X 12-Core CPU, 64GBs of RAM and a NVIDIA RTX 2070 GPU (8GB). The training took approximately 9.5 hours for 500 epochs, using batch size 256 and learning rate .0005.

We explore our framework’s capabilities through a series of experiments. More specifically, we perform a sanity test, and then test the level to which simulated trajectories that were generated by P2C predicted behaviors resemble real trajectories captured in videos. We also compare P2C with existing frameworks, specifically RVO, a CCP “baseline”, and Random Walk ( [12], [37], [48])); we use the same RVO parameters as the one used in the underlying parameter model (Section 3.1), and the CCP “baseline” refers to inputting CCP weights that fit the video best according to intuition; we list these values Section 4.2. Therefore, we quantitatively evaluate P2C comparing with real data and other baseline models to assess plausibility and statistical similarity to real behaviors. Finally, we qualitatively evaluate our results via a user study (Section 4.3) that is designed to assess predictive power, realism and behavior diversity. Please refer to the supplemented video for animated versions of the results.

4.1 Preliminary Experiments

Before testing P2C against other frameworks and real data, we conduct a sanity check to confirm P2C is successful in handling dominants behaviors individually. More specifically, we generate five sets of simulated trajectories that correspond to specific behavior profiles and therefore distinct behaviors; these are goal-only, group-only, interaction-only and goal-seeking with maximum and minimum connectivity $\{(0,0,0,0), (0,0,0,0.3), (0,0,0,1), (0,0,0,1), (0,0,0,1), (0,0,0,0)\}$ are the values for the respective profiles). Having these profiles we generate 100 simulations for each of the five experiment types, and run the P2C model on the constructed image encodings to obtain the predicted profiles. For each experiment type, we average the profile predictions over the 100 samples and display the results in the form of color-coded heatmaps in Figure 8; we expect that the predicted profiles will roughly match the inputted ones. For the first three experiments we keep the connectivity constant, whereas for the last two we keep the set of $\{w_g, w_g, w_i\}$ fixed.

The generated heatmaps (Figure 8) reveal that in all five scenarios, P2C clearly predicts the dominant behaviors and produces crowd profiles close to the expected numbers. The observation that no value reaches 1 or 0 can be attributed to the fact that agents exhibits diverse behaviors, and environmental influences like obstacle reforms and high-densities can affect the ability of agents to execute the “ideal” behavior; these situations indirectly affect the prediction capabilities of our model. Still, the sanity check results reveal promising predictions for real-data trajectories.

4.2 Quantitative Evaluation

We compare with real-world data to assess behavior similarity with P2C, and for that we utilize the three datasets: Students, Zara, and Church. For each video, we use the real agent trajectories to construct the corresponding encoded images; each representing a certain area and frame range. Then, we get the predicted crowd parameters from our trained model, simulate the crowd behavior, and document the new trajectories. Figure 9 provides visual correspondence of real and simulated trajectories with the sample underlying profile; P2C was used on the real data to predict the crowd profile which was then used to generate three simulations with different spawn and goal positions. We observe that all runs are close to real data behaviors in terms of stationary/moving percentage and nature of trajectories.

Testing on Real Data

In order to quantitatively assess our framework, we examine four metrics: Density, Speed, Distance to Nearest Neighbour (DNN),
and Movement Direction Diversity (MDD); the last is based on the frequency of different movement angles derived from consecutive position vectors. Essentially, these metrics are used to quantify the behavior similarity between real and simulated movements. Mainly, the agents in the videos walk, turn, and stand still. Combinations of these movements reveal their behavior, for example, if agents were walking and then stopped for a long period of time, then they are likely to be grouping, or inspecting a nearby POI. Conversely, if two agents move constantly while having high DNN, they possibly have goal-dominant behaviors with high connectivity value. Hence, if two populations (real and P2C-generated) have similar distributions of these selected metrics, we believe it is a representative measure of behavior similarity. We visualize said distributions corresponding to the real and simulated trajectories in Figure 10.

These box-plot pairs, essentially reveal the distribution summaries which are used as comparison measures to assess whether the simulated trajectories are consistent enough with the real data. Figure 10 shows that the overall trend of the real distributions is compatible with the simulated ones, especially in median and interquartile range (IQR) [36]. That said, the similarity of the box-plot whiskers is partly subjective; they provide a simplified global view of the data so we can not claim with absolute certainty faithfulness to real behaviors.

**Comparisons**

We carry out further experiments to evaluate the quality of our framework compared to existing “baselines”. Specifically, we use additional three behavior models for comparisons, (1) RVO simulator, (2) Random Walk, and (3) the CCP baseline of having a single manually-defined profile across all data. For the latter, we used the universal profile of \(\{.45,.35,.2,.25\}\) since we believe it to be the most common profile in our three datasets. A more careful choice of profiles for each dataset, each frame range and each area could be made, however we would have to integrate an effectiveness metric, as the process would be time-consuming and trial-and-error based. To make our deductions more robust, we calculate the Jensen-Shannon Divergence (JSD) [33] between the distributions of each metric type as calculated from the real trajectories and the per-model simulated trajectories. JSD is a variation of the KL-divergence and is bounded by 0 and 1 with values closer to 0 suggesting similar probability distributions.

Table 3 presents the JSD between the real and simulated permetric distributions, when using various clips from all available datasets; we use 95 seconds in total. P2C has the lowest JSD score for three out of four metrics suggesting that it could serve as the new baseline or at least be comparable with the CCP baseline and RVO.

Table: Real Data Faithfulness Model Comparison.

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Density</th>
<th>Velocity</th>
<th>DNN</th>
<th>Direction Diversity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>JSD↓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2C</td>
<td>0.0167</td>
<td>0.173</td>
<td>0.0355</td>
<td>0.0137</td>
</tr>
<tr>
<td>CCP baseline</td>
<td>0.0189</td>
<td>0.315</td>
<td>0.0421</td>
<td>0.0113</td>
</tr>
<tr>
<td>RVO</td>
<td>0.0171</td>
<td>0.483</td>
<td>0.0753</td>
<td>0.0753</td>
</tr>
<tr>
<td>Random Walk</td>
<td>0.0651</td>
<td>0.461</td>
<td>0.246</td>
<td></td>
</tr>
</tbody>
</table>

In order to investigate our work further, we build the Fundamental Diagram (FD) comparing the trend of real-world and simulated data. We combine the Students and Zara datasets resulting to a of total 562 agent trajectories and 9.5 minutes of simulation; the generated diagram is shown in Figure 11. The local density \(d_a^t\) of an agent \(a\) at time \(t\) is calculated using the equation by Helbing et al. [17], where \(p_a^t\) and \(N(a^t)\) are the position and the set of all neighbors of agent \(a\), in a radius \(R_a = 3.6m\) at time \(t\) respectively,

\[
d_a^t = \sum_{i \in N(a^t)} \frac{1}{R_a^2} \exp(-||p_a^t - p_i^t||^2/R_a^2).
\]

The FD reveals a higher similarity between the real-world and P2C simulations, compared to RVO and CCP baseline. The latter methods seem to face a flow decrease sooner, as the density increases.

![Fundamental Diagram](image)

**Fig. 11: Fundamental Diagram for Real-World, P2C, CCP baseline and RVO data. We combine the Students and Zara datasets.**

For a more qualitative comparison, in Figure 12, we visually isolate two agents (non-grey paths) and make them move according to each of the eligible systems. Here the objective is not to have identical trajectories per set, but rather exhibit similar behaviors spatially and temporally. For the dotted agent, P2C is closest to the real data both speed-wise and path-wise. For the other agent (continuous line), we emphasize that the CCP baseline moves faster and more goal-oriented, whereas P2C and real make similar deceleration and stops and wonder through the environment more.
4.3 User Study

4.3.1 Setup

We release a 20’ user study with 44 participants to assess our framework’s simulations quality and believability. More specifically, we design the study to test two hypotheses:

- \( H_1 \): The behaviors generated by P2C are realistic-looking to users.
- \( H_2 \): P2C can be considered state-of-the-art in this task.

To do that, we split our study in three parts. Initially, we exhibit ten videos, each depicting either real or P2C trajectories in random order, and we ask participants to choose which they consider more realistic; this is used to support \( H_1 \). Then, we show four separate scene-specific videos for three scenes: Church Yard, Commercial Street (Zara), and Campus. For each one, we first show the real video, hinting on the observed behaviors in specific scenarios, and then simulated videos of the real, P2C, and RVO-generated trajectories in random order e.g., RVO, Real, P2C; we ask participants to rank behavior similarity with the real video in a 7-point Like-rt scale. Finally, we directly compare P2C with RVO by showing five pairs of (side-by-side) videos and asking users to choose the video that exhibits behaviors most likely to be seen in real life; the latter two parts are used to asses \( H_2 \). Figure 13 provides details on simulation visualization. Our participants were diverse in age (> 18, < 65), sex (25M/17F), country of work (i.e., Cyprus, France, Spain, and Germany). Overall, the users’ crowd simulation knowledge varied with an average of 2.5/5; we believe this to be a relatively high score for a no prerequisite study. We asked participants to view the videos in their entirety and judge based on overall behaviors rather than frame by frame paths.

4.3.2 Findings

For the first part of the study, we measure how confused the users were when having to distinguish real versus P2C trajectories. We summarise the results from the ten pairs in a boxplot (Figure 14-left), from which we can deduce that approximately half the time users were unable to differentiate the real paths. It is important to note that this was not due to a balanced proportion of nearly correct and nearly wrong identifications across the ten sample pairs, as might be suggested by an average result. Instead, on a per-sample basis, users were equally confused in around 50% of the cases. For example, in sample 2, approximately 54.5% of users misidentified the path shown as “real” when it was, in fact, generated by P2C. This collective confusion across all samples suggests that users encountered difficulty distinguishing between the real and P2C-generated paths. Consequently, it implies an inability to confidently discern between the two, providing preliminary support for hypothesis \( H_1 \).

To gather more evidence about \( H_2 \), we directly compare P2C with RVO. Figure 14-right shows the results of this third user study part. We see that for the five examples (grey boxplot) the users on average prefer P2C paths. We find that the two examples for which users (on average) prefer RVO are the ones coming from the Zara video; we claim again, this is logical since this particular video does not contain many complex behaviors i.e.,
mostly goal-seeking and hence RVO struggles less to depict them; most existing works are goal-seeking and avoidance simulators, failing to handle a wider range of behaviors. So, we also illustrate the results when removing the Zara samples and see a significant increase in the preference scores in favour of P2C. Still, we deduce that even though P2C is preferable on average, it is not a clear winner and thus RVO is a good baseline to use for these kinds of comparisons. Regarding $H_2$, there is some evidence to support our claim but we note that it is not strong. We observe that user perception is deeply biased by scenario context, agents’ modelling, quality of animations (if any), and degrees-of-freedom, hence more robust quantitative metrics are needed capable of evaluating behavior similarity in virtual crowds.

5 Discussion

In conclusion, this paper introduces the Paths-to-Crowds framework, a novel tool that maps encoded images from input videos capturing diverse crowd behaviors to intuitive crowd parameters. The significance of our framework lies in its ability to parameterize crowd data within the selected parameter space, providing valuable control, explainability, and generalizability in diverse and custom environments without compromising simulation plausibility. As shown in our results, P2C empowers authors to synthesize simulations by implicitly describing desired crowd behaviors extracted from trajectories, serving as a versatile tool for crowd behavior simulation. In addition, it provides enhanced usability, enabling users to generate novel crowds through an interactive authoring tool. We demonstrated that with this tool, users can create custom virtual environments with behavioral patterns defined by reference videos, facilitating population according to user preferences.

We performed a comprehensive quantitative evaluation of our trained model, demonstrating the authenticity and realism of our generated simulation behaviors through the use of distribution similarity metrics and diagrams. This evaluation includes comparisons with real behaviors and alternative baselines, including RVO, CCP “baseline”, and Random Walk. Our qualitative evaluation was carried out via a user study and indicates that P2C is capable of capturing more complex behaviors compared to the other methods tested and closely mirrors real crowd behaviors in terms of naturalness.

Limitations and Future Work

Despite the capabilities of our framework, the design and evaluation stages reveal certain limitations. Primarily, the underlying behavior model we employ has limitations that accumulate in our framework, as CCP was not trained on real data. Our primary goal is to parameterize behaviors in any video; however, the current framework relies on videos with supplementary tracked data to generate channel encodings. Consequently, P2C does not support in-the-wild videos, as it requires accompanying agent trajectories. Moreover, most videos that are experiencing trajectory constraints often exhibit common behaviors, posing challenges in fully evaluating our framework’s capabilities. This underscores the necessity for crowd datasets featuring more complex behaviors.

Our future objectives involve integrating a video tracking component into our pipeline to create a system that supports any video, eliminating the requirement for tracked positions as input. Another interesting improvement would involve allowing designated behavior areas for a single prediction to overlap, enabling an averaging metric of multiple predictions. This approach would result in a more robust and representative profile. Additionally, our current system treats behavior areas as fixed-sized rectangular segments. In the future, we aim to explore integrating more complex area shapes and automating their establishment based on real, observed environmental information.

Acknowledgments

This work has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska Curie grant agreement No 860768 (CLIPE project), the European Union’s Horizon 2020 Research and Innovation Programme under Grant Agreement No 739578, and the Government of the Republic of Cyprus through the Deputy Ministry of Research, Innovation and Digital Policy.

References

Marilena Lemonari is a Marie-Sklodowska Curie ESR fellow and a PhD student at the Department of Computer Science, University of Cyprus. Marilena received a BSc in Mathematics (honors) at the University of Nottingham, and an MSc in Machine Learning (honors) from University College London (UCL). Her main research interests include topics in machine learning, crowd simulation, and control. Contact her at lemonari.marilena@ucy.ac.cy.

Andreas Panayiotou is a PhD candidate in Computer Science at the University of Cyprus and a Research Associate at CYENS - Centre of Excellence. He holds a BSc and an MSc in Computer Science from Department of Computer Science, University of Cyprus. His research focuses on crowd simulation, machine learning, virtual environments, and computer animation. Contact him at a.panayiotou@cyens.org.cy.

Nuria Pelechano is an Associate Professor at the Universitat Politècnica de Catalunya, Spain, where she is a member of the Research Center for Visualization, Virtual Reality and Graphics Interaction (ViRVIG). She is the president of the Eurographics Spanish Chapter. Nuria obtained her PhD from the University of Pennsylvania and a Masters degree with honors from the University College of London. She participates in several EU projects (MSCA ITN-ETN CLIP, HORIZON-101093159-XR4ED), and Spanish projects. Her research interests include simulation and animation of virtual humans and crowds, and human-avatar interaction in Virtual Reality. Contact her at npelechano@cs.upc.edu.

Theodoros Kyriakou is a PhD candidate in Computer Science at the University of Cyprus and a Research Associate at CYENS - Centre of Excellence. Theodoros has a BSc and an MSc in Computer Science from the University of Cyprus. His main research focuses on virtual instrument performances, character animation, machine learning, and virtual environments. Contact him at t.kyriakou@cyens.org.cy.

Yiorgos Chrysanthou is the Research Director of the Centre on Interactive Media, Smart Systems and emerging Technologies - CYENS Centre of Excellence. He is also a Professor at the Computer Science Department of the University of Cyprus. Yiorgos was educated in the UK (BSc and PhD from Queen Mary College, University of London) and worked for several years as a research fellow and a lecturer at University College London. He published over 80 papers in journals and international conferences on computer graphics and virtual reality and served as the local or overall coordinator of over 25 research projects. His research interests lie in the general area of 3D Computer Graphics, recently focusing more on the development of algorithms for real-time AR and VR rendering, reconstruction of urban environments and computer animation. Contact him at y.chrysanthou@cyens.org.cy.

Andreas Aristidou is an Assistant Professor at the Department of Computer Science, University of Cyprus. He had been a Cambridge European Trust fellow at the University of Cambridge, where he obtained his PhD. Andreas has a BSc in Informatics and Telecommunications from the National and Kapodistrian University of Athens and he is an honor graduate of Kings College London. His primary research focuses on computer graphics and vision, with specialized expertise in character animation and virtual humans. His research extends to digital cultural heritage, exploring VR/AR/XR environments, while it investigates the applications of Conformal Geometric Algebra in the realm of computer graphics. Contact him at a.aristidou@ieee.org.

Panayiotis Charalambous is the team leader of the V-EUPNEA: Living, Breathing Virtual Worlds group at CYENS Centre of Excellence since June 2019 and is a Research Fellow at The Cyprus Institute. He was educated at the National and Kapodistrian University of Athens, Greece (BSc and MSc) and has a PhD from the University of Cyprus. Panayiotis worked as a Visiting Lecturer in the Department of Computer Science at the University of Cyprus and as an Associate Research Scientist and a Computational Scientist at The Cyprus Institute. Between 2014-2016 he was a post-doctoral fellow at INRIA Rennes, France. Panayiotis has coordinated and worked on several projects funded by Horizon 2020, RPF/RIF and other agencies. He is a member of the ACM and the Eurographics association and has served in the program committee of several conferences and is a reviewer of several prestigious journals and conferences. His research interests include Computer Animation, Virtual Humans, Crowd Simulation, Digital Cultural Heritage and Real-Time Technologies for Digital Twins and Filming. Contact him at p.charalambous@cyens.org.cy.