Semantic Synthesis of Pedestrian Locomotion Supplementary Material

Maria Priisalu[1], Ciprian Paduraru[2,3], Aleksis Pirinen[1], and Cristian Sminchisescu[1,3,4]

[1] Department of Mathematics, Faculty of Engineering, Lund University
[2] The Research Institute of the University of Bucharest (ICUB), Romania
[3] Institute of Mathematics of the Romanian Academy
[4] Google Research
{maria.priisalu,aleksis.pirinen, cristian.sminchisescu}@math.lth.se
ciprian.paduraru@fmi.unibuc.ro

1 Supplementary Videos and Further Visualizations

In the file `pedestrian_synthesis_in_waymo.mp4`, example trajectories generated by our goal-free SPL agent on the Waymo test set are shown. The agent is visualized as a skeleton, which is generated by the human locomotion network (HLN), cf. §2.3 in the main paper. Also, 3d bounding boxes of cars and other pedestrians are shown. The agent can be seen to wait for a bus before crossing and starting to run when cars close to it start moving. The agent can also be seen crossing the street while walking behind a car and crossing a busy intersection. When the ground truth trajectories end, the bounding boxes of cars and pedestrians stand still, but the agent continues to produce plausible trajectories. In Fig. 1 a small qualitative example of the agent cutting corners and leaving the crosswalk can be seen. Such plausible but perhaps imperfect behaviour from the point of view of traffic laws would need to be explicitly modelled to be achieved by classical planners (based on methods such as visibility graphs or spatial navigation meshes).

In the file `pedestrian_synthesis_in_cityscapes.mp4`, example trajectories generated by our SPL-goal agent on the Cityscapes test set are shown. Two observed pedestrians visualized as the blue and yellow bounding box can be seen walking along the pavement. A car is approaching marked by the red bounding box. It is unclear where the pedestrians are heading, as they may decide to cross the street on the crosswalk, continue along the curb or continue forward. Our agent can be used to rollout the motion of the pedestrian in yellow in all of these cases. The agent is initialized with the past movements of the pedestrian in yellow after which it generates movement in multiple plausible directions, shown as the moving skeleton (which again is produced by the HLN module). In the second scene we show the temporal visualization of Fig. 5 from the main paper.

Frame by frame visualizations from the CARLA test set in Fig. 3 shows that the HLN produces smooth poses even when accelerating from standing to running or when running away from cars.

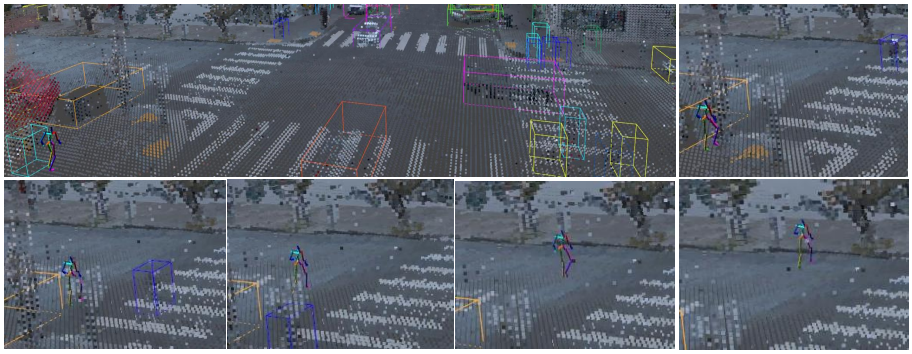

Fig. 1: A subsampled pose sequence in Waymo showing the SPL agent cutting corners and choosing to avoid the crosswalk. This kind of imperfect but plausible traffic behaviour would not be modelled by classical planning methods.To the *top left:* it can be seen that no cars are approaching the pedestrian. The pedestrian continues to the left once on the pavement.

Further visualizations of the SPL agent on the CARLA test set can be seen in Fig. 2. In these visualization we show the SPL agent trained only with RL, i.e. without any alternation between imitation learning and RL (cf. Algorithm 1).

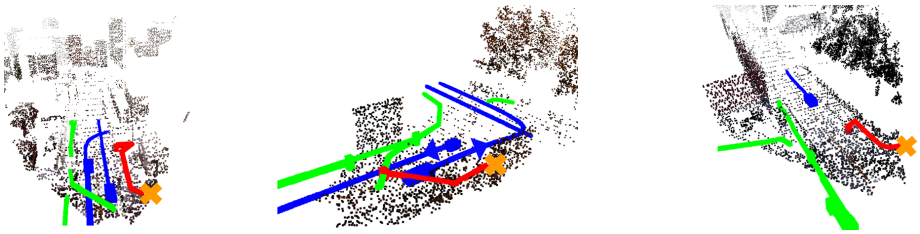

Fig. 2: Additional 1-minute trajectories of our goal-free SPL agent on the CARLA test. Note that in these examples we show an SPL agent that was trained without any objective that encourages it to track another pedestrian – it was trained only to walk safely and plausibly in the environment based on the reward signal in §2.4 of the main paper. Car and person trajectories are shown in blue and green, respectively. Red indicates the agent's trajectory, with terminal position indicated by an orange cross. *Left:* Agent walking along a pavement. The pavement also continues to the right close to the initial location, and the agent initially walks a few steps in that direction. It then then turns to move towards the camera, which indicates the inherent multi-modality of plausible paths to take. *Middle:* When initialized on the road on a trajectory of a crossing pedestrian, the agent moves away from the road onto the pavement parallel to to the road. Blue arrows indicate the directions of the nearby cars. Note that the agent is not hit by any of the cars (the bottom car has moved away before the agent reaches that trajectory). The agent produces a plausible but different trajectory from the one it was initialized on. *Right:* Agent moving to the sidewalk when initialized on the road. The agent follows the curvature of the pavement in the end of its trajectory.

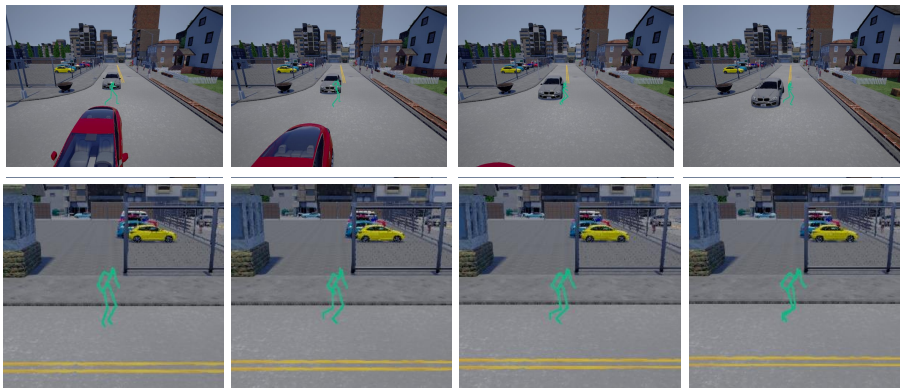

Fig. 3: *Top:* A subsampled (every 4th frame) pose sequence in the CARLA test set showing the agent successfully running to avoid two cars. *Bottom:* The agent starts running from a standing initial position to leave the road. The movements are temporally smooth even when the agent accelerates from standing.

2 Details of our Model Architecture

In Fig. 5 we show the network architecture of the proposed model. The semantic and RGB top-view projections as well as the dynamic occupancy map (constructed as shown in Fig. 4) are processed by the convolutional features extractor, which consists of two convolutional layers with ReLU activation and followed by max pooling. The agent's past trajectories (i.e. the agent history) are processed by an LSTM. Finally, two independent fully connected layers process the HLN's locomotion feature $\boldsymbol{h}_{t-1}$, the agent's displacement $\boldsymbol{d}_t$ to the closest vehicle, the encoded agent history $\boldsymbol{f}_t$, and the convolutional features to produce two feature vectors $\boldsymbol{f}_u$ and $\boldsymbol{f}_\mu$. The multinomial distribution over unit directions is given by $\pi_\Theta(\boldsymbol{U}_t|\boldsymbol{s}_t) = \text{softmax}(\boldsymbol{f}_u + \boldsymbol{u}_{t-1})$. The addition of $\boldsymbol{u}_{t-1}$ acts as a prior to make the agent move in the same general direction unless motivated by the current state $\boldsymbol{s}_t$ to change direction. The mean speed is given by $\mu_t = \sigma(\boldsymbol{f}_u)$, where σ is the sigmoid function. Finally, the agent's velocity $\boldsymbol{v}_t$ for the current timestep is given by $\boldsymbol{v}_t = |\boldsymbol{v}_t|\boldsymbol{u}_t$.

3 Semantic 3d Reconstruction of Cityscapes

The 3d reconstruction of the environment E_t is performed in two parts – one global reconstruction for static objects S and a frame-by-frame stereo reconstruction for moving objects in D_t. Semantic segmentation is used to mask out people, cars and bikes, when performing sparse 3d reconstruction. Dynamic objects are represented by cuboids in D_t, which are found by performing 2d object detection followed by triangulation.

A semantic segmentation network is used to estimate the semantic mask of each frame. The points in the found 2d semantic masks are triangulated to find the corresponding 3d points in the world. A 3d point $\boldsymbol{p_i}$ which is visible in

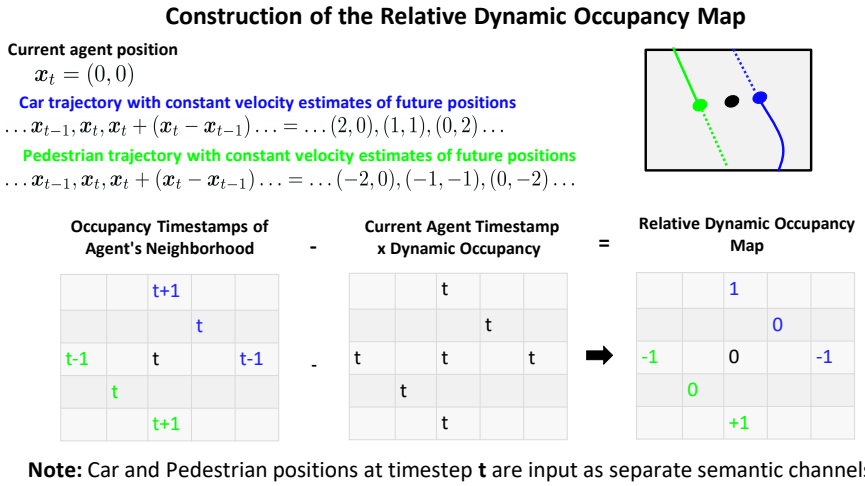

Fig. 4: Construction of the dynamic occupancy map. The pedestrian and car trajectories are extended by constant velocity estimates at each timestep, as indicated by the dashed lines in the top-right rectangle. Timestamps of future and past occupancy in the agent's neighborhood are then mapped into an occupancy map, centered around the agent. Finally, the current timestamp t is subtracted to provide a relative dynamic occupancy map centered around 0. Note that the pedestrians and cars present are represented in separate channels in the semantic 3d reconstruction input E.

frames $i, .., j$ has the 2d projections $\boldsymbol{u_i}, .., \boldsymbol{u_j}$. Each 2d point $\boldsymbol{u_i}$ corresponds to a semantic label estimate l_i. The mode of $l_i, ..., l_j$ is assigned as the label of p_i. Alternatively, a 3d semantic segmentation method could be used. The resulting pointclouds are regularized by a voxel grid G. In the regularization, the color of a voxel is determined by the mode color of the points in the voxel. Semantic class is determined in the same fashion. The voxelized static environment is filtered with a median filter with size $(1, 4, 4)$. Finally, the height of the ground plane is estimated as the mode of points in the 3d semantic pointcloud with the semantic labels for ground, road, sidewalk, parking, terrain and rail tracks.

4 Training Procedure for the SPL and SPL-goal Agents

The SPL and SPL-goal agents are trained to simultaneously minimize the pedestrian trajectory forecasting error by maximizing the model likelihood on the pedestrian trajectories in the data, and to maximize the respective reward functions. In §2 of the main paper it is shown that the two objectives can be rewritten as a single one, with two sampling techniques (sampling agent locations and moves from the expert trajectories, or from the current policy). Given expert trajectories $(\boldsymbol{x}_0^i, \ldots \boldsymbol{x}_T^i) \in \mathcal{D}$, which can also be described by the state-velocity

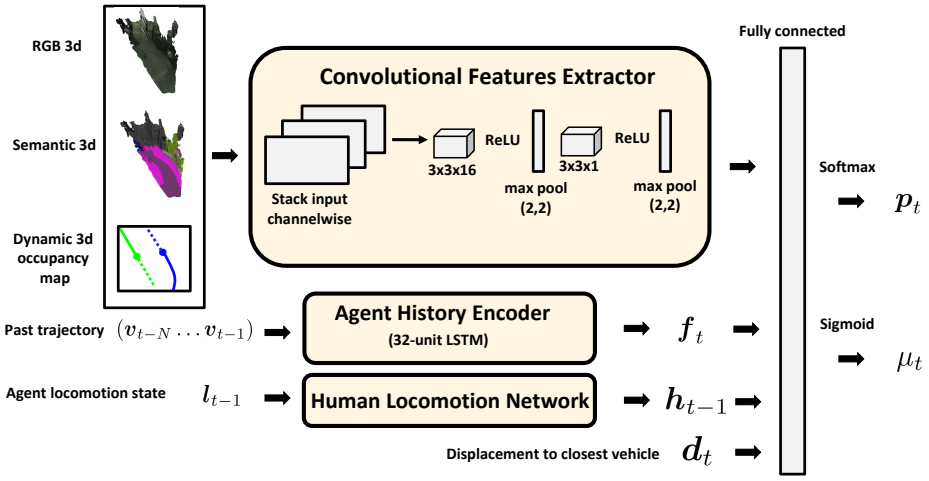

Fig. 5: Semantic trajectory policy network (STPN) architecture. A top-view of an agent-centered local crop of the 3d pointcloud with RGB and semantic labels, together with a top-view of the dynamic occupancy map, is processed by the convolutional features extractor. The N last agent velocities $\boldsymbol{v}_{t-N}, \dots, \boldsymbol{v}_{t-1}$ are fed to a 32-unit LSTM, whose hidden state is denoted $\boldsymbol{f}_t$. The human locomotion network (HLN) extracts from the previous locomotion state $\boldsymbol{l}_{t-1}$ the locomotion feature $\boldsymbol{h}_{t-1}$ (recall that the STPN is executed *before* the HLN, and thus the STPN only has access to the *previous* hidden state $\boldsymbol{h}_{t-1}$ of the HLN). Finally, the convolutional features, agent history feature $\boldsymbol{f}_t$, locomotion feature $\boldsymbol{h}_{t-1}$ and displacement vector $\boldsymbol{d}_t$ to the closest vehicle are concatenated and fed to two separate fully connected layers. The first layer feeds into a softmax to produce a distribution $\boldsymbol{p}_t$ over the unit movement directions (unit velocities), and the second layer feeds into a sigmoid to produce the mean speed μ_t for a normal distribution from which a speed $|\boldsymbol{v}_t|$ is sampled. The agent's velocity $\boldsymbol{v}_t$ for the current timestep is finally given by $\boldsymbol{v}_t = |\boldsymbol{v}_t|\boldsymbol{u}_t$.

pairs $(\boldsymbol{s}_0^i, \boldsymbol{v}_0^i) \dots (\boldsymbol{s}_T^i, \boldsymbol{v}_T^i)$, and given the set of agent initializations[5] $\mathcal{I}$, the full optimization can be performed by following Algorithm 1 that samples gradients from the expert pedestrian (initialized from $(\boldsymbol{x}, v) \in \mathcal{D}$) and the policy trajectories (initialized from $\boldsymbol{x}_0 \in \mathcal{I}$, and thereafter following $\boldsymbol{v}_t \sim \pi_\Theta(\boldsymbol{v}|\boldsymbol{s}_t)$ where $\boldsymbol{x}_{t+1}$ is given by HLN step towards $\boldsymbol{x}_t + \boldsymbol{v}_t$), respectively.

In all experiments the semantic trajectory policy network (STPN, cf §2.2 in the main paper) is trained first without the HLN, cf. Table 1. The SPL agent is then refined with the HLN executing the steps provided by the STPN, as visualized in Fig. 3 of the main paper. During this training step the HLN weights are kept frozen (please refer to §2.5 for training procedure and evaluation). The CARLA goal driven agent SPL-goal-CARLA (Table 1 in main) is initialized from

[5] $\mathcal{I}$ contains initializations near cars, near pedestrians, on pavement and at random.

Algorithm 1 Unified Batch Policy Gradient and Maximum Likelihood Estimation

for N epochs **do**
  for pedestrian i in $\mathcal{D}$ **do**
    Initialize batch gradient direction $\boldsymbol{g} = 0$
    Initialize agent on the ith pedestrian's initial state $\boldsymbol{s}_0 = \boldsymbol{s}_0^i$
    for t in $[0, T]$ **do**
      Take expert action $\boldsymbol{v} = \boldsymbol{v}_t^i$
      Evaluate gradient $\boldsymbol{g} = \boldsymbol{g} + \partial_\Theta R(\boldsymbol{s}, \boldsymbol{v}) \log(\pi_\Theta(\boldsymbol{s}|\boldsymbol{v}))$
    end for
    for Initialization j in $\mathcal{I}$ **do**
      Initialize agent on the jth initial state $\boldsymbol{s} = \boldsymbol{s}_0^j$
      for t in $[0, T]$ **do**
        Sample action $\boldsymbol{v}_t^j \sim \pi_\Theta(\boldsymbol{v}|\boldsymbol{s})$
        Take action $\boldsymbol{v} = \boldsymbol{v}_t^j$
        Evaluate gradient $\boldsymbol{g} = \boldsymbol{g} + \partial_\Theta R(\boldsymbol{s}, \boldsymbol{v}) \log(\pi_\Theta(\boldsymbol{v}|\boldsymbol{s}))$
      end for
    end for
    Update Θ with gradient step in direction $\boldsymbol{g}$.
  end for
end for

the pretrained weights of SPL-CARLA noted here for clarity as STPN-CARLA to indicate that no training with the HLN is performed. Since the Cityscapes dataset does not contain tracking and therefore trajectories, the STPN-Cityscapes is initialized from STPN-CARLA, and is trained on the Cityscapes dataset for 14 epochs. The STPN-Cityscapes is the initialization of the SPL-goal-Cityscapes presented in Table 3. Finally the SPL Waymo agent in Table 3 is initialized from the weights of STPN Waymo. The initialization and training epochs are gathered in Table 1. The agent history is initialized randomly when the agent is not initialized on top of a pedestrian in the data. This provides a random initial direction and promotes exploration in early stages of learning.

Table 1: Number of epochs trained for different models presented. The number of epochs of pretraining shows the number of epochs STPL is trained without the HLN, and No of epochs is the number of epochs the two modules are trained together.

Model	Dataset	Weight initialization	No. of epochs pretraining	No. of epochs training
SPL	CARLA	Random	10	8
SPL-goal	CARLA	(STPN-CARLA)	21	14
SPL	Waymo	Random	1	1
STPN	Cityscapes	(STPN-CARLA)	14	-
SPL-goal	Cityscapes	(STPN-Cityscapes)	22	17

5 Ablation Study of Reward Function

Table 2: Ablations of the STPN with an average over all of the presented initializations. The ablation results are on the CARLA validation set with a maximal episode length of 5.8s. f_d is the frequency on locations occupied by pedestrians, f_{pav} is the frequency on pavement. The full model balances between collisions, travelling far and implicitly learns to stay on pavement without explicitly staying on pedestrian trajectories.

Model	$f_{o,p}$	f_{car}	d	f_d	f_{pav}
$R_{dist} + R_{coll}$	0.01	0	4.7	0.14	0.61
R_{ped}	0.55	0	9.6	0.24	0.48
$R_{coll} + R_d$	0.03	0.1	13	0.45	0.79
$R_{coll} + R_k$	0.02	0	16	0.04	0.45
$R_{coll} + R_{ped}, \lambda_p = \lambda_v = \lambda_s$	0.04	0	15	0.31	0.38
$R_{coll} + R_{ped}$	0.04	0	8.5	0.32	0.77

To show the need for the different reward components we present a short ablation study in table Table 2 on the CARLA test set. The different models are the STPLN agent trained with the listed reward weights. The R_{ped} is trained with a positive reward for being on pedestrian trajectory $\lambda_d = 0.01$ and $\lambda_k = 0.01$. This leads to an agent that is oblivious to collisions and collides with a frequency of 0.55. As a baseline, we present an agent trained without the pedestrian reward components and instead a positive reward $R_{dist} + R_{coll}$ for moving further from the start location and a negative reward for collisions ($\lambda_{dist} = 0.001$). This agent $R_{dist} + R_{coll}$ visits the pedestrian occupancy map seldom $\boldsymbol{f}_d = 0.14$. Further only including one of the pedestrian reward components $R_{coll} + R_d$,$R_{coll} + R_k$ leads to an agent that ignores cars when only rewarded for staying on expert trajectories, and to an agent that seldom visits the pedestrian occupancy map when only rewarded for staying on the pedestrian heatmap. Finally the the model trained with the full reward $R_{coll} + R_{ped}$ balances between all of these metrics. To show the effect of the different collision components of the reward, we set the penalty for colliding with cars, pedestrians and obstacles equal to 1 $R_{coll} + R_{ped}, \lambda_p = \lambda_v = \lambda_s$ and observe that this leads to an agent that is on pavementf_{pav} half of the times compared to the proposed model $R_{coll} + R_{ped}$.