EUROGRAPHICS 2019 / P. Alliez and F. Pellacini
(Guest Editors)

Volume 38 (2019), Number 2

Supplementary Material

He Wang[1†], Sören Pirk[1†], Ersin Yumer[2], Vladimir G. Kim[3], Ozan Sener[4], Srinath Sridhar[1], and Leonidas J. Guibas[1]

[1]Stanford University, [2]Uber ATG, [3]Adobe Research, [4]Intel Labs

Abstract

In this supplementary material, we provide additional details for the method proposed in the paper. We describe the explicit representation of the interaction sequences which we used to generate interactions in Section 1. We also provide details on the formal definition and estimation of Gaussian Mixture Models(GMMs) which are used to represent low level motions of objects. Moreover, we show the statistics of our collected video data and explain how the data is annotated. Finally, we provide the numbers of physical constraint violations as a quantitative measure of Action Plot RNN generation results.

1. Action File

The final result of our method is an animation that is either reconstructed or generated. In this section, we explain the format for storing interaction sequences generated by our algorithm. It can be used to directly forward action sequences to the animation module or to store the sequences for offline purposes. We create a text file, representing the generated animations and call it *action file*. Each action file is composed of two sections; the initial scene description and the interaction sequence.

To initialize scenes, we store the locations of objects (in table coordinate space) and their states. We represent action plots as a sequence of actions in the action file. Each line in the interaction sequence corresponds to an action that may cause changes of the object arrangement or their states. We store the type of an action (e.g. pour), the start and end time, the active object, and possible a receiving object. Some actions might not include receiving objects, like for opening a book. In this case the receiving object is denoted as *undefined*. The whole entry for one action includes the class name, class index, start state, end state, and start and end positions for each object. We do not include the start and end position of the receiving object as we define it as static for the period of the performed action.

Fig 1 shows an example of an action file. We also visualize the part of the animation generated by the file in Fig 2.

We generate action files for two different purposes: reconstruction and generation. For the training videos, action files are automatically generated using the video processing module explained in Section 5 of the paper. For generated interaction sequences, action files are generated as an output from our action plot model.

```
#object, index, position x, position y, state
bowl, 0, 6.17324710021, 22.3913006964, 0
orange, 0, 16.0965807984, 22.6723686561, 0
orange, 1, 13.6330087587, 18.3454133162, 0
orange, 2, 18.236141743, 15.3901778681, 0
orange, 3, 20.814936995, 18.7214436402, 0

# action, obj1, obj id1, obj2, obj id2,
# ... startTime, endTime, [start x, start y,
# ... end x, end y, start state, end state]x2
idle, undef, 0, undef, 0, 0.0, 4.8, 0, 0, 0, 0, 0, 0, 0, 0
move, hand, 0, undef, 0, 4.8, 6.4, 0, 0,18,15, 0, 0, 0, 0
move, orange, 2, bowl, 0, 6.4, 7.3, 18,15, 6, 22, 0, 1, 0, 1
move, hand, 0, undef, 0, 7.3, 7.5, 6, 22, 0, 0, 0, 0, 0, 0
```

Figure 1: *An example action file*

2. Object Position Gaussian Mixture Models

The target position of a moving object, without a specific receiving object, depends on the following spatial and temporal factors: the class of the object, the spatial arrangement of all the other objects, and the dynamics of the action, like speed and duration. We model each factor as a Gaussian Mixture Model and multiply them to obtain the final proposal distribution following the assumption that each factor is independent.

Object Class GMM $\mathbf{P_{c_i}}$ The object class factor, represented in the form of a GMM, captures the distribution of the 2-D position of an object in table coordinate space solely based on the object class c_i independent of other objects. To train the GMM we use the tracked positions of all objects of a class in our dataset and apply the expectation-maximization (EM) algorithm.

Since we detect each object as a bounding box, we need to choose a point within the bounding box to represent its position.

† Equal contribution.

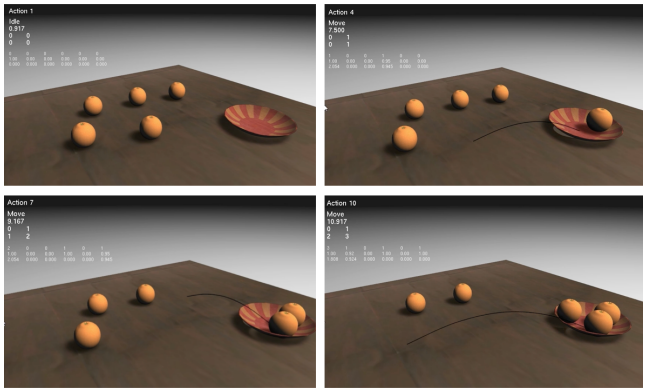

Figure 2: *The updated scenes corresponding to the actions in Fig. 1. Top left: animation scene at the end of action 1, {idle} (same to initial scene); top right: animation scene at the end of action 4, {move, orange, 2, bowl, 0}; bottom left: animation scene at the end of action 7, {move, orange, 4, bowl, 0}; bottom right: animation scene at the end of action 10, {move, orange, 0, bowl, 0}.*

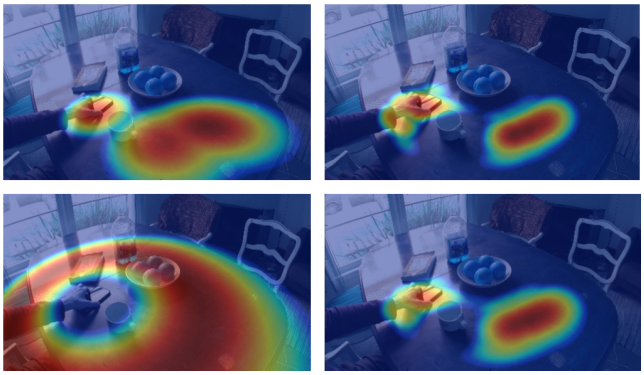

Figure 3: *Log-likehood of the proposal distribution for the target position of the interacted phone. Top left: Log-likelihood corresponding to the object class factor,* $\log P_{c_i}(x)$*; top right: the log-likelihood corresponding to the spatial arrangements,* $\sum_{j \neq i} \log P_{c_i,c_j}$*; bottom left: log-likelihood corresponding to the speed and duration,* $\log P_{speed,c_i}(x)$*; bottom right: log-likelihood of the final proposal distribution, obtained via multiplying aforementioned distributions, used to sample a new position for the phone.*

Although the center is a natural choice, it is error prone since the bounding box estimate is not always perfect. Therefore, we use data augmentation and add additional random data points around the center corresponding to the uncertainty in bounding box estimation. We add 50 more points by adding a random Gaussian noise with variance of 1/4 of the bounding box height.

Finally, based on the observation of the data point distribution, the number of components used in Gaussian Mixture is selected using the AIC metric between 2 and 4.

Intra/Inter-Class Relative Distance GMM, $\mathbf{P_{c_i,c_j}}$ A intra/inter-class factor captures the relative spatial distribution of objects based on their classes. It models both, the affinity of certain classes, like cups and bottles, as well as repulsion motivated by the collision avoidance. Similar to the object class factor, we also use data augmentation as well as an AIC metric to select the number of Gaussian components between 2 and 4.

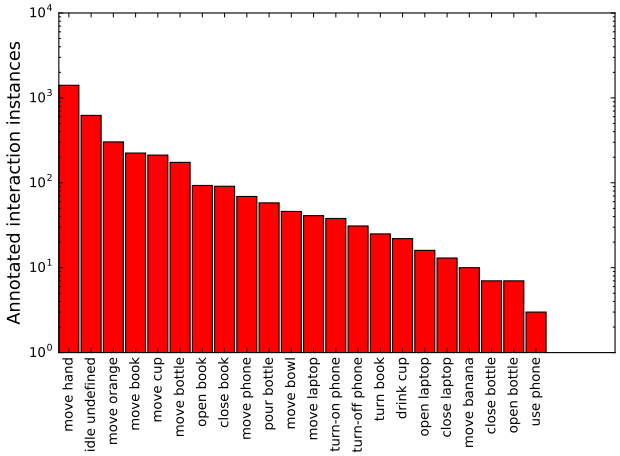

Figure 4: *Number of interaction instances for every type in our dataset.*

Speed GMM, $\mathbf{P_{c_i,speed}}$ The speed factor captures the dynamics of each object class in terms of the magnitude of the average speed obtained during the action. Since the points chosen as a position within the bounding box typically do not change during the interaction, we do not need the data augmentation for this model. Error in estimation of the center cancels out, since average speed computation includes subtraction of the start from the end position. We also use a single Gaussian.

3. Video Annotations and Data Statistics

Action segmentation is commonly referred to solve the segmentation problem for every action in time and classify each constituent segment. We manually annotate the action segmentation for all of 75 video clips in our dataset. For annotation consistency, a single person from our authors annotated all the video sequences. The beginning time of any action is always the moment that the object starts moving while the end of action is the moment that the motion stops. "Idle" action is bound by the end time of the previous action and the start time of the next action. Our dataset contains 22 different types of interactions. The number of each interaction type annotated in our dataset is shown in Fig.4.

4. Quantitative Evaluation of Action Plot RNN Generation

Evaluating the generated animation is an open problem since there is no metric which can capture the quality of the animation directly. Using the ground truth information is not desired, as we want our algorithm to deviate from the training data and generate as many plausible and consistent animations as possible.

Although there is no perfect metric on the quality of the gen-

Table 1: *Constraints on the two action plots $\mathbf{L}_{t-1}$ and $\mathbf{L}_t$ in terms of predicates defining prerequisites and effects. Here Op.[o] represents actions that operate on a single object, e.g. turn on phone, read book, while Op.[o_1, o_2] represents actions that operate on two objects, e.g. pour bottle to cup.*

Action	Prerequisites	Consequences
Idle		$o_t = o_{t-1}$, $s_t = s_{t-1}$
Move [o_t]	$o_{t-1} = o_t$ or None	$s_t = s_{t-1}$
Op. [o_t]	$o_{t-1} = o_t$	update s_t
Op. [$o_{t,1}$, $o_{t,2}$]	$o_{t-1,1} = o_{t,1}$	update $s_{t,1}$, $s_{t,2}$

erated sequences, the generated sequence needs to follow physical laws and constraints. In other words, the generated sequence should be physically plausible. For example, the generated sequence should not move an orange from an empty bowl, or should not move a bottle without first grasping it. Although it is intractable to list all such physical holistically, it is possible to list them for a small set of actions and objects. Moreover, such approaches have been widely used in semantic planning [KS92] literature since action planning has similar requirements. We define set of prerequisites per action class in terms of predicates of the current active object o_{t-1} and object states s_{t-1}. Furthermore, we also define the effect of the action on the world as well as states after the action o_{t-1}, s_t. We require o_{t-1} and s_{t-1} to satisfy the predicate on the prerequisites, and o_t and s_t to satisfy the predicates on the effects. We list the used predicates of physical laws in Table 1.

We measure the plausibility of our generated sequences by counting the number of violations of the prerequisites in Table 1. By counting the number of errors in a sequence of 100 generated actions of 100 action plots, we get an average error rate of $1.36\% \pm 1.09\%$.

References

[KS92] KAUTZ H., SELMAN B.: Planning as satisfiability. In *Proceedings of the 10th European Conference on Artificial Intelligence* (New York, NY, USA, 1992), ECAI '92, John Wiley & Sons, Inc., pp. 359–363. 3