Detecting Anomalies in Object Appearance and Motion Dynamics

by

Mazen Alotaibi

A PROJECT

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science in Artificial Intelligence

Presented August 29, 2022
Commencement June 2023

ABSTRACT OF THE PROJECT OF

Mazen Alotaibi for the degree of Master of Science presented on August 29, 2022. Title: Detecting Anomalies in Object Appearance and Motion Dynamics

Machine common sense remains a broad, potentially unbounded problem in AI. Our focus is to move toward AI systems that can develop common-sense reasoning similar to humans to detect anomalies. In particular, we study the problem of detecting the violation of expectations when object appearance or motion dynamics change from simulated experiments. We have developed a system of multiple components to solve the problem of machine common sense. The system contains three main components: a two-stage tracker, a role assigner, and a rule-based reasoning agent. We evaluate our system on scenes from the DARPA Machine Common Sense, Passive Violation of Expectation task.

Corresponding e-mail address: alotaima@oregonstate.edu

Detecting Anomalies in Object Appearance and Motion Dynamics

Mazen Alotaibi, *Oregon State University*
Chanho Kim, *Oregon State University*
Zeyad Shureih, *Oregon State University*
Ashish Malik, *Oregon State University*
Jed Irvine, *Oregon State University*
Stefan Lee, *Oregon State University*
Fuxin Li, *Oregon State University*
Alan Fern, *Oregon State University*

***Abstract*—Machine common sense remains a broad, potentially unbounded problem in AI. Our focus is to move toward AI systems that can develop common-sense reasoning similar to humans to detect anomalies. In particular, we study the problem of detecting the violation of expectations when object appearance or motion dynamics change from simulated experiments. We have developed a system of multiple components to solve the problem of machine common sense. The system contains three main components: a two-stage tracker, a role assigner, and a rule-based reasoning agent. We evaluate our system on scenes from the DARPA Machine Common Sense, Passive Violation of Expectation task.**

I. Introduction

When an adult plays Peekaboo with an infant, although the adult is in front of the infant and the adult's body is visible, the infant is surprised that you appeared when you uncover your face. Children younger than four years old can't determine that objects can't appear or disappear without an explanation because they haven't developed the cognitive ability to understand that objects and people still exist even when they can't see or hear them. This is one of the types of common sense reasoning, and it is called Object Permanence. However, in humans, the cognitive ability to understand and develop common sense reasoning is developed throughout the person's life. Despite impressive progress in artificial intelligence in many domains, artificial intelligence systems are still far from human performance regarding common sense reasoning about objects in the world. Artificial Intelligence systems don't learn common sense reasoning automatically, and they need to be forced to understand those concepts to consider them.

An example of the importance of Common Sense developed within Machines is in self-driving cars. Let us assume that an Oregon State University student wants to leave campus to go home after a long day of hard work. The student owns a self-driving car that will always take them to their home safely. Corvallis had heavy rain on that day, and the usual route from the OSU campus to the student's house was filled with water, and the GPS and other indicators of the blocked route weren't updated for the self-driving car to be aware of the flooding that blocked the road. The self-driving car has an objective to reach the final destination. However, the route is blocked, and visual systems can observe this. The dilemma for the self-driving car is whether it should continue its usual route as it has never experienced this situation before, or should it find a different, safer route? Although the developers of the self-driving car can hard-code logic to solve a lot of situations where a machine can encounter similar cases, it would require a lot of engineering effort and might not consider unseen and rare situations. Therefore, developing common sense ability is essential for the machines to make the right choices without being hard-coded to take them.

To build artificial intelligence systems to learn common sense reasoning, we will need to define machine common sense and quantify development progress. Machine Common Sense is the ability of machines to understand their world, behave reasonably in unforeseen situations, communicate naturally with people, and learn from new experiences [1]. A benchmark [2] was created to develop artificial intelligence systems to quantify the progress of machine common sense development. The benchmark measures the performance of the intelligent agent perceiving the world without any interaction on many common sense reasoning tasks, such as understanding gravity force, object permanence, and other cases. The benchmark expects the agent to detect anomalies when objects violate expectations.

In this paper, we have presented an artificial intelligence system to perceive the scene to solve machine common sense tasks in the benchmark. The system contains three main parts: a two-stage tracker to track objects in the scene, a role assigner to label objects for their role in the video sequence, and a rule-based reasoning agent to output the final prediction to determine if the given video sequence violates a set of expectations.

II. Background

The benchmark [2] has the intelligent agent to perceive the world, and the agent can't interact with objects in the world. The two major components are: how the agent can track objects in the world and how it can reason that there is a violation of expectations when it happens.

There are two types of Multi-Object Tracking algorithms to track objects in a sequence of frames: Batch and Online methods. Batch methods [3] [4] process past and future frames to determine the object identities in the current frame. Whereas Online methods [5] [6] [7] process past frames without knowing future frames to determine the object identities in the current frame. Online methods tend to have lower performance in exchange for real-time tracking. On the other hand, Batch methods are meant to maximize performance but are slow.

For developing reasoning agents, there are two prior works [8] [9] that utilized a physical simulation module that predicts the future object states from current beliefs. For the first paper [8], one of the solved tasks using their method was predicting the future position of a set of stacked objects of whether they will fall or not. The method estimates whether the stacked objects would fall and in which direction they would fall. The estimated values would be used as the input's expected behavior. If the simulation resulted in different values than what has been estimated with a certain degree of freedom, the method would raise a violation of expectations. For the other paper [9], they solved the problem of estimating the motion dynamics of objects in the scene that can be occluded for a short or long time. Their method is based on developing object beliefs of the object's future position as the expected object beliefs. The prior object beliefs will be updated if the observed object position varies within a reasonable range. Otherwise, if the observed object position varies a lot, the method would raise a violation of expectations. Both methods rely on a physical simulation module to estimate the future object states based on current beliefs.

III. Dataset

The benchmark is defined as a task where a stationary agent observes a simulated scene of two types of objects. First, focus objects that have free motion, and the agent will be tasked to track those objects because the change of object appearance and motion dynamics occur on focus objects. Second, environmental objects that occlude or support focus objects. When a scene contains a focus object that violates the scene type's expectations, it is labeled as an implausible scene. Otherwise, it is labeled as a plausible scene.

There are five types of scenes with their assigned expectations:

1) **Object Permanence (OP)**: Focus objects can't appear or disappear from the scene without an explanation.
2) **Gravity Support (GRAV)**: A dropped focus object follows gravity.
3) **Collisions (COLL)**: Focus objects can't change their motion or appearance without an explanation.
4) **Shape Constancy (SC)**: Focus objects can't change their appearance.
5) **Spatiotemporal Continuity (STC)**: A thrown focus object needs to have a continuous motion.

The benchmark provides an Integrated Learning Environment (ILE) that we utilize to generate data to train and evaluate our system. As shown in Figure 2, the inputs to the system are the following for every video sequence: RGB frames, ground-truth segmentation masks, and depth images. The expected output of our approach is a binary classification of whether the video sequence is plausible or not based on a set of expectations that the system developed.

IV. Approach

For this section, we present our approach, which involves five modules. The system is composed of multiple components that are either algorithm, learning, or rule-based components, as it is shown in Figure 3. First, **Mask to 2dbox** module expects a segmentation mask as an input and outputs a set of 2D bounding boxes. The module utilizes an algorithm to convert the segmentation mask of every object in a scene into 2D bounding boxes for every object in the scene.

*A. **Tracker***

To track objects in a given sequence of frames, we need to utilize a tracking module, which takes the history of 2D bounding boxes of all objects in the scene and the RGB frames of the scene as an input and outputs a set of tracks for every tracked object in the scene. For our *Tracker* module, we implemented a Two-stage Tracker as it is shown in Figure 4.

*1) **First-stage of the tracker:*** It expects the history of 2D bounding boxes of all objects in the scene and the RGB frames of the scene as an input and outputs a set of continuous tracklets for every tracked object without fragments in the scene. Tracklets are tracks with short lengths, and continuous tracklets are short tracks without fragments in their tracks. For example, as is shown in Figure 5, the first stage of the tracker will construct two different tracklets for the same object before and after occlusion. For the first stage of the tracker, we utilized an out-of-the-box online tracker that relied on motion and appearance cues. The tracker contains a features extractor, ResNet50 [10], and Bilinear LSTM [5] as recurrent network to learn the task of multi-object tracking.

One of the issues of using online trackers is that the tracker would make mistakes when the new detection of an object that is partially visible and has similar features to a previous tracklet, so the tracker would merge both based on appearance cues. However, when the object is fully visible after it merges the partially visual detection with a tracklet, the tracker will append the later detections to the tracklet because they share the same motion cues. In other words, the tracker would make a mistake due to the limited amount of

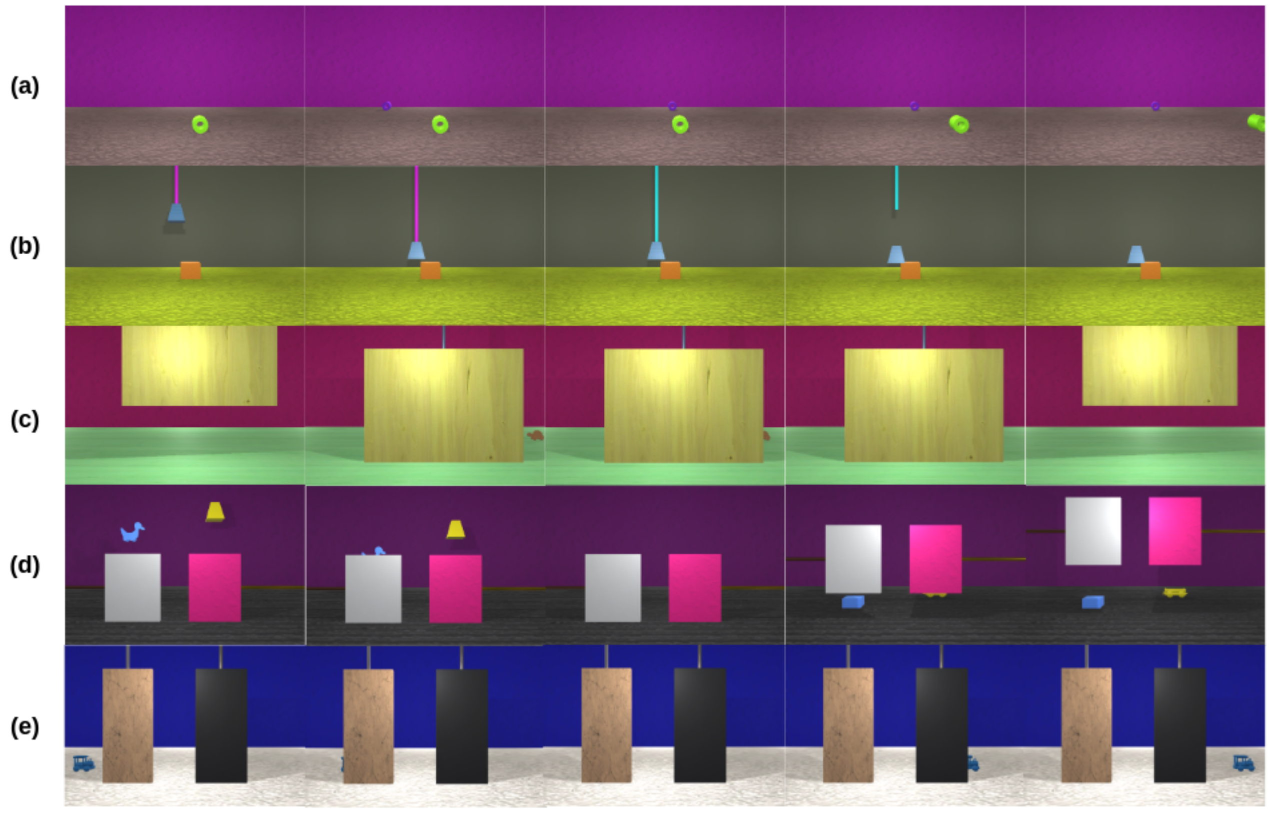

Fig. 1. Five types of Machine Common Sense scenes: a) Collision (COLL) b) Gravity Support (GRAV) c) Object Permanence (OP) d) Shape Constancy (SC) e) Spatiotemporal Continuity (STC)

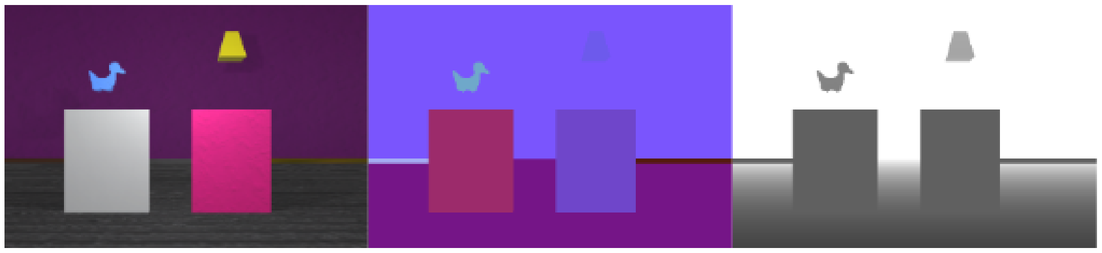

Fig. 2. An example of input passed to the system where the left image corresponds to an RGB frame, the middle image corresponds to a ground-truth segmentation mask and the right image to a depth image.

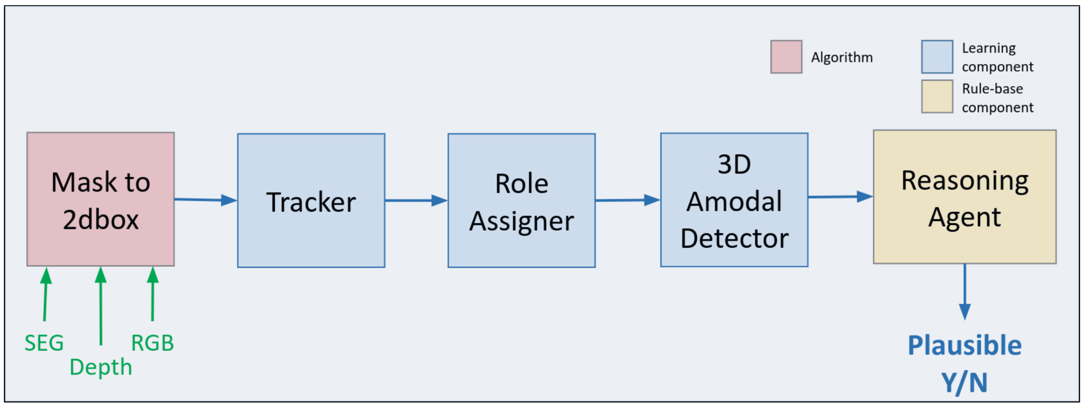

Fig. 3. For every video, the system takes a sequence of RGB, segmentation masks, and depth information for every frame in a video. From left to right, the system converts segmentation masks into 2D bounding boxes for every object in every frame, builds tracks for every object, assigns a role for whether a track is a focus object or not, estimates the 3D position of the object, then predict if the scene is plausible or not.

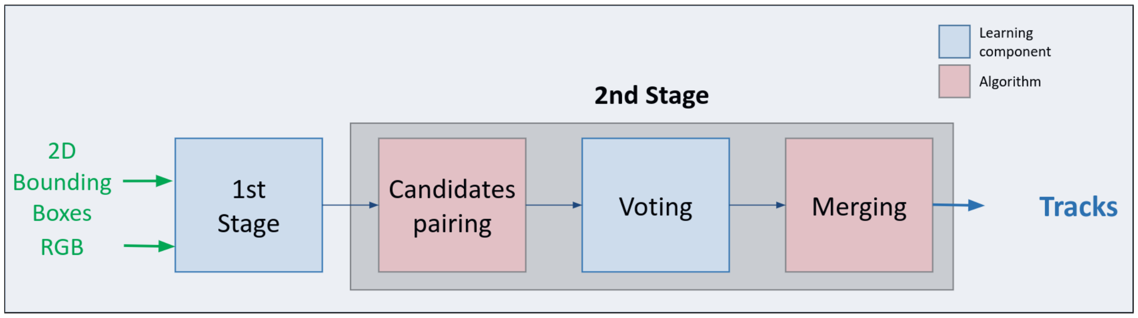

Fig. 4. For every video, the tracker takes the history of 2D bounding boxes for all objects in the scene and the RGB frames. From left to right

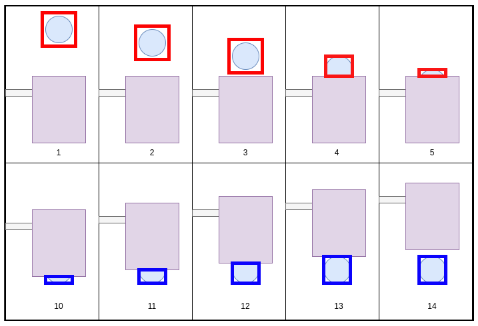

Fig. 5. This is an example of the output of the first stage of the tracker, where it creates two separate tracklets because of the occlusion. The red bounding boxes are for before occlusion, and the blue ones are for after occlusion.

information visible to it, but when it has all of the required information to correct its previous mistakes, it doesn't. For example, as shown in Figure 5, when the online tracker is shown detection on frame 10, the tracker would match this new detection with the tracklets created from 1 to 5 because they share some motion and object appearance cues. However, when the object is fully visible on frame 14, and it is visible that the object shape is drastically different, the tracker won't correct the tracklet based on this disagreement because the tracker relied on motion cues after the mistake was committed.

Although online trackers allow us to develop real-time trackers, they tend to be less accurate, as shown before. In the Machine Common Sense challenge, we care more about the performance of the tracker than its speed. Therefore, we have introduced a second stage of the tracker that will process the entire video sequence and merge tracklets that were split due to short or long occlusions.

*2) **Second-stage of the tracker**:* It expects a set of continuous tracklets in the scene and the RGB frames of the scene as an input and outputs a collection of tracks for every tracked object in the scene. The second stage of the tracker has three components: Candidates pairing, Voting, and Merging.

First, the **candidates pairing** component creates a set of candidate pairs where a pair is an excellent pair to be merged according to simple rules. For a given potential pair, the first tracklet is initialized before the second tracklet, and both tracklets can't coexist in the same frame simultaneously.

Second, the **voting** component computes a similarity score module on multiple selected pairs of two different tracklets to generate a set of similarity scores. If the average of the set of similarity scores is above a threshold, both tracklets will be considered potential merge candidates. We utilized a learnable model for the similarity score module that takes two images and outputs their similarity score. For the learn-able model is Siamese Network [11] with ResNet18 [10], pre-trained on ImageNet [12] and fine-tuned on Machine Common Sense dataset, as its backbone.

Finally, the **merging** component merges potential merge candidates based on simple rules. For given potential merge candidates, those candidates shouldn't coexist when a chain of tracklets is merged, and if there is a coexist, the module would be the best candidate based on the voting score.

B. *Role Assigner and 3D Amodal Detector*

In our reasoning agent module, the module depends on the objects' roles and their 3D position in the scene to classify if the scene is plausible or implausible. Hence, we have **Role Assigner** module to assign roles and **3D Amodal Detector** module to estimate the 3D position of every object in the scene.

*1) **Role Assigner**:* It takes a set of tracks in the scene and outputs a binary classification of whether the track is for a focus object or not. We utilized a learnable model that relies on the appearance cues of the objects for every track. The learn-able model is composed of a ResNet18 [10], pre-trained on ImageNet [12] and fine-tuned on Machine Common Sense Dataset, as its backbone and multiple linear layers.

*2) **3D Amodal Detector**:* It takes a set of tracks in the scene and the depth information for every frame. The module outputs the 3D position of every object in the scene in the world coordinate. We utilized an out-of-the-box pre-trained model, CenterTrack [13], to estimate the 3D position of the objects in the scene.

*C. **Rule-based Reasoning Agent***

We developed a simple algorithm for the reasoning agent module that utilized assumptions to detect violation-of-expectations that works well based on our experiments. The module uses different assumptions for every scene type. For each scene type, we will introduce the scene's set of expectations, a set of assumptions, and a set of rules to detect violations of expectations.

*1) **Object Permanence (OP):*** The expectation for OP scenes is focus object can't appear or disappear from the scene without an explanation. We formulated two assumptions for focus objects: 1. they can only enter a scene from the edges of the frame. 2) Once they leave the scene, they can't re-appear. We developed three rules to detect implausibility. First, check if all focus objects entered the scene from any of the sides of the frame. Second, check if the object left the scene. If it did and there wasn't any re-appearing of the same object, this scene is plausible. Third, if the object didn't leave the scene, check if the object was in the last scene and was expected to be behind an occluder wall. If it was, then the scene is plausible. Otherwise, the scene is implausible.

*2) **Gravity Support (GRAV):*** The expectation for GRAV scenes is that a dropped focus object follows gravity. We formulated two assumptions for a fallen focus object: 1) has only gravity force applied to it. 2) It is supported by the support object if it lands on top of it and is well-balanced. We developed a rule to detect plausibility. Given the first position of the fallen focus object and the position of the support object, estimate the last relative position of the fallen object to set an expectation. In other words, based on the first position of the fallen focus object and the position of the support object, will the fallen focus object be on top of the support object or the ground at the end of the video sequence? If we expect the fallen focus object to be on top of the support object, but it wasn't, then we raise a violation of expectation. Otherwise, it will raise that this scene is plausible. On the other hand, if we expect the fallen focus object to be on the ground, then we raise a violation of expectation if the last position of the fallen focus object wasn't on the floor.

*3) **Collisions (COLL):*** The expectation for the COLL scene is that focus objects can't change their motion dynamics or object appearance without an explanation. We formulated two assumptions for focus objects: 1) they can only enter a scene from the edges of the frame. 2) Two focus objects can only interact with each other if they intersect at a certain point in time and space. We developed two rules to detect plausibility. First, check if all focus objects entered the scene from any of the sides of the frame. Second, check if two objects share the same intersect at a certain point of time and space. The scene is plausible if two cases occurred. First, both objects share the same intersect at a certain point of time and space, then interact. Second, they don't share the same intersect at a certain point of time and space, and they don't interact. Otherwise, the scene is implausible.

*4) **Shape Constancy (SC):*** The expectation for SC scenes is that focus objects can't change their object appearance. We formulated two assumptions for focus objects: 1) they can only enter a scene from the edges of the frame. 2) they don't change their object appearance. We developed a single rule to detect implausibility. If any of the tracks was initialized in the middle of the scene, this scene is implausible.

*5) **Spatiotemporal Continuity (STC):*** The expectation for STC scenes is that a thrown focus object must have continuous motion. We formulated two assumptions for focus objects: 1) they can only enter a scene from the edges of the frame. 2) they have continuous motion, meaning they don't have gaps without explanation. We developed three rules to detect plausibility. First, check if all focus objects entered the scene from any of the sides of the frame. Second, check if there are fragments in the focus object tracks. If there aren't fragments, this scene is plausible. Third, if there are fragments, check the start and end positions when the object disappeared because it was occluded. The scene is plausible if it was occluded throughout the entire fragment space. If it wasn't fully occluded throughout the fragment as the whole space, the scene is implausible.

V. Evaluation

We evaluated our components on unseen generated data from the MCS program evaluation set. We randomly sampled 100 scenes of plausible and implausible scenes for each scene type. In other words, for every scene, we have 50 randomly sampled scenes that are plausible and another 50 randomly sampled scenes that are implausible. We compared every component against a baseline and the reasoning agent component against a baseline with actual data input from the system pipeline.

We didn't evaluate two components: the Mask to 2dbbox converter because it is a trivial task, and the 3D Amodal Detector is an out-of-the-box model.

*A. **Tracker***

We evaluated our tracker against a baseline that uses ResNet18, which was pre-trained on the ImageNet [12] dataset but wasn't fine-tuned on the Machine Common Sense dataset as the backbone of the Siamese network in the second stage. Because the baseline uses a model that wasn't trained for the given dataset, we normalized the extracted features for both images from the backbone. We applied the dot product of the normalized vectors.

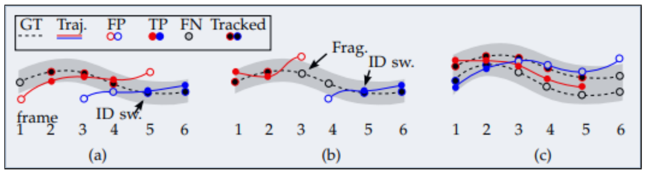

Fig. 6. For cases (a) and (b), the tracker assigns a new identity to the same track, which counts as an identity switch. However, for the case of (c), two tracks overlap, so we don't count them as an identity switch.

For the metrics, we utilize Number of Identity Switch (IDSW) and Multiple Object Tracking Accuracy (MOTA) as two metrics to compare both models [14]. IDSW measures the number of times the tracker changes a track's ID. In Figure 6, (a) and (b) show two types of mistakes the tracker could make when the tracker makes a single identity switch by assigning a new identity (blue) for a track that had already been assigned an identity (red). The difference between (a) and (b) is that the tracker in (a) made the identity switch without a fragment, whereas the tracker (b) made the identity switch with a fragment. As for the last case (c), the IDSW metric won't consider it an identity switch because two tracks overlap. Although we use ground truth segmentation masks, which give us ground truth bounding boxes, we utilize MOTA as it accounts for IDSW and deleted tracklets or detections. In other words, the MOTA score helps us quantify the tracker's overall performance.

$$MOTA = 1 - \frac{\sum_t (FN_t + FP_t + IDSW_t)}{\sum_t GT_t}$$

TABLE I
TRACKERS PERFORMANCE ON EVERY SCENE TYPE BASED ON IDSW AND MOTA SCORE.

	Baseline		Ours	
	IDSW↓	MOTA↑	IDSW↓	MOTA↑
Collisions	0.1	0.9944	0.1	0.9944
Object Permanence	**0.14**	**0.9966**	0.18	0.9961
Shape Constancy	0.44	0.9867	**0.29**	**0.9891**
Spatiotemporal Continuity	0.0	0.9924	0.0	0.9924
Gravity Support	0.0	0.9998	0.0	0.9998
All	0.136	0.9939	**0.114**	**0.9943**

As shown in Table I, our tracker performed well overall. However, for Object Permanence, the tracker made more IDSW than the baseline, which reduced the MOTA score. In Object Permanence scenes, objects appear and disappear within implausible scenes. In case an object appears again after leaving the scene, the tracker makes the mistake of assigning the exact identity of the object even though the object has already left the scene. Although the tracker failed to split tracks, the reasoning agent has the mechanism to raise a violation of expectations when this case happens. On the other hand, our tracker outperformed the baseline for Shape Constancy scenes. It means that the tracker managed to distinguish between objects that change their shapes but keep their colors. However, our tracker still made mistakes when the object changed to a similar shape or changed its scale.

B. *Role Assigner*

We evaluated our role assigner against the decision tree model, which classifies roles based on the object's area and position in the frame. We evaluated every scene type based on the F1-score for focus objects as positive examples for the metric.

As shown in Table II, our role assigner outperformed the baseline in every scene type. Our role assigner made mistakes in Shape Constancy and Gravity Support scenes. For the Shape Constancy scenes, our role assigner made mistakes when the occluder pole was small, and the assigner misclassified the occluder pole as a focus object. The assigner misclassified specific fallen objects for Gravity Support scenes as environmental objects because their shapes seemed similar to occluder walls.

C. *Reasoning Agent*

We evaluated our reasoning agent against a baseline learnable model that was trained on plausible scenes, as shown in Figure **??**. For every timestep, the baseline generates context for each focus object using a transformer [15] based encoder, combines contextual information with focus object information and feeds it to a Recurrent Neural Network (RNN) [16], and generates N future step predictions. We evaluated every scene type based on the F1 score for implausible scenes as positive examples for the metric.

As shown in Table III, our reasoning agent's overall performance is better than the baseline model. However, the baseline outperformed our reasoning agent on Collision scenes, and this is due to the design of the rules for Collision scenes. Our reasoning agent performed better than the baseline model for the other types of scenes.

As shown in Table IV, our reasoning agent's overall performance is better than the baseline model. Our reasoning agent performance when given actual tracks is similar when given ground-truth tracks besides in two scene types. For Collisions scenes, our reasoning agent wasn't robust to noise in the pipeline that it predicted more implausible scenes than plausible ones. As for Object Permanence scenes, the reduction in F1-score from 94% to 75% was due to the tracker.

VI. FUTURE WORK

Our approach has much room for improved improvements from data generation to training models by considering cases where they failed. We will introduce the problem, explain the reason why the issue existed, and suggest solutions to

TABLE II

Role Assigners performance on every scene type based on F1-score for focus objects as positive examples.

	Baseline			Ours		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Collisions	0.71	0.82	0.76	1.0	1.0	**1.0**
Object Permanence	0.6	1.0	0.75	1.0	1.0	**1.0**
Shape Constancy	0.47	1.0	0.64	1.0	0.98	**0.99**
Spatiotemporal Continuity	0.42	1.0	0.6	1.0	1.0	**1.0**
Gravity Support	1.0	0.75	0.86	1.0	0.96	**0.98**
All			0.72			**0.99**

TABLE III

Reasoning Agents' performance given ground truth as an input on every scene type based on F1-score for implausible scenes as positive examples.

	Baseline			Ours		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Collisions	0.54	1.0	**0.70**	0.29	0.16	0.21
Object Permanence	0.47	0.88	0.61	1.0	0.88	**0.94**
Shape Constancy	0.0	0.0	0.0	1.0	1.0	**1.0**
Spatiotemporal Continuity	0.56	1.0	0.72	1.0	0.84	**0.91**
Gravity Support	0.0	0.0	0.0	1.0	0.92	**0.96**
All			0.41			**0.8**

TABLE IV

Reasoning Agents' performance given actual input from the pipeline on every scene type based on F1-score for implausible scenes as positive examples.

	Baseline			Ours		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Collisions	0.5	1.0	**0.67**	0.45	0.64	0.53
Object Permanence	0.5	1.0	0.67	1.0	0.6	**0.75**
Shape Constancy	0.0	0.0	0.0	1.0	0.86	**0.92**
Spatiotemporal Continuity	0.55	1.0	0.7	1.0	0.84	**0.91**
Gravity Support	0.0	0.0	0.0	1.0	0.92	**0.95**
All			0.41			**0.82**

resolve the issue to improve the system's overall performance in upcoming work.

A. *Data Generation*

For training data, we needed to generate data using configuration scripts with many knobs to change without any reference point to determine if our data was decent or not. Therefore, we needed to manually inspect the generated data to determine if they were ready to be fed to our models to be trained. However, we have many data to go through, and data generation takes time.

We suggest that generating high-quality data will enhance most learnable components' performance. High-quality data indicate the data is well-representative of the evaluation data distribution, reduction of simulation errors, and many variations in visual and motion cues.

B. *Tracker*

As for the tracker, we have room for improvement in both the first and second stages of the tracker.

The first stage of the tracker made a single type of mistake. When the object changes its color without any occlusion, the tracker assigns the same object identity even though the object changes its color drastically. The tracker couldn't segment the track into two different tracklets because the first stage of the tracker relied on motion cues more than appearance cues in the case.

To improve the first stage of the tracker, we have two options. First, training the first-stage tracker on these cases where the object changes its appearance without occlusion. Second, introducing a module to scan the appearance of objects in tracks to determine if the object changed its appearance or not. If the object changes its appearance, the module will split the track into two tracklets.

For the second stage of the tracker, the component failed in two cases. The first case is when the focused object changes to a similar shape and keeps the same color. The component failed because it relied on color cues more than object shape cues. This could be resolved by training on hard examples where the object changes its shape to a similar shape and keeps its color. The second case is when the object changes its size. The component sometimes makes mistakes in connecting those two tracklets, even though they are incredibly different sizes. The component makes this mistake because visual cues of the two objects, before and after occlusion, are resized to fixed image size, making the component unaware of the change in object scale. We must consider the object's scale in our similarity scoring to resolve this issue.

C. Reasoning Agent

As we have seen in the rule-based reasoning agent, the rules weren't robust to noise in the pipeline when giving ground truth data versus actual data. We suggest design rules that are more robust to noise in the pipeline.

REFERENCES

[1] D. Gunning, "Machine common sense concept paper," 2018.

[2] R. Riochet, M. Y. Castro, M. Bernard, A. Lerer, R. Fergus, V. Izard, and E. Dupoux, "Intphys: A framework and benchmark for visual intuitive physics reasoning," 2018. [Online]. Available: https://arxiv.org/abs/1803.07616

[3] K. Zhao, T. Imaseki, H. Mouri, E. Suzuki, and T. Matsukawa, "From certain to uncertain: Toward optimal solution for offline multiple object tracking," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, pp. 2506–2513.

[4] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European Conference Computer Vision (ECCV)*, 2016.

[5] C. Kim, L. Fuxin, M. Alotaibi, and J. M. Rehg, "Discriminative appearance modeling with multi-track pooling for real-time multi-object tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 9553–9562.

[6] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky, "Bot-sort: Robust associations multi-pedestrian tracking," 2022. [Online]. Available: https://arxiv.org/abs/2206.14651

[7] P. Chu, J. Wang, Q. You, H. Ling, and Z. Liu, "Transmot: Spatial-temporal graph transformer for multiple object tracking," 2021. [Online]. Available: https://arxiv.org/abs/2104.00194

[8] P. W. Battaglia, J. B. Hamrick, and J. B. Tenenbaum, "Simulation as an engine of physical scene understanding," *Proceedings of the National Academy of Sciences*, vol. 110, no. 45, pp. 18 327–18 332, 2013. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.1306572110

[9] K. Smith, L. Mei, S. Yao, J. Wu, E. Spelke, J. Tenenbaum, and T. Ullman, "Modeling expectation violation in intuitive physics with coarse probabilistic object representations," *Advances in neural information processing systems*, vol. 32, 2019.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[11] G. Koch, R. Zemel, R. Salakhutdinov *et al.*, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2. Lille, 2015, p. 0.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[13] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points," *ECCV*, 2020.

[14] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "Motchallenge 2015: Towards a benchmark for multi-target tracking," 2015. [Online]. Available: https://arxiv.org/abs/1504.01942

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[16] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.