

Compositional Video Prediction

by Yufei Ye, Maneesh Singh, Abhinav Gupta, and Shubham Tulsiani

Presented by: Minh Pham

Overview

- Introduction
- Related Works
- Proposed Method
- Experiments and Results



Introduction

- Video prediction:

Introduction

- Video prediction:
  - From a single image, generate a video

UCF

Introduction

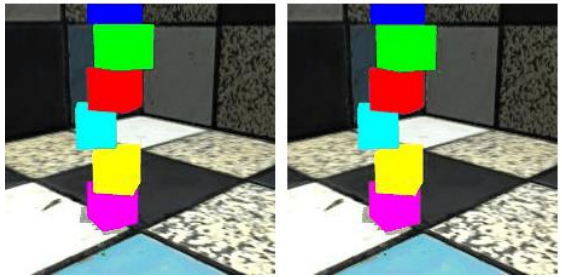
- Video prediction:
  - From a single image, generate a video
  - A prediction of future states

Introduction

- Video prediction:
  - From a single image, generate a video
  - A prediction of future states
- Issues:

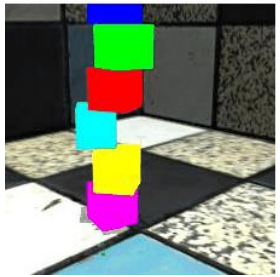
Introduction

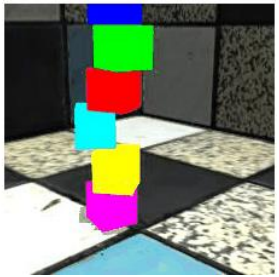
- Video prediction:
  - From a single image, generate a video
  - A prediction of future states
- Issues:
  - Handling multiple objects in frame

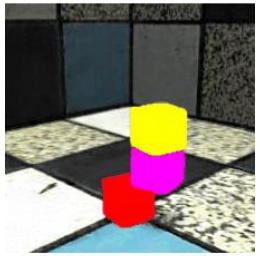


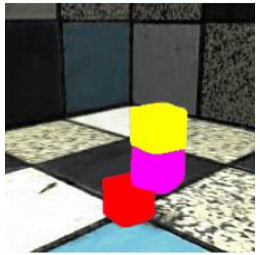
Introduction

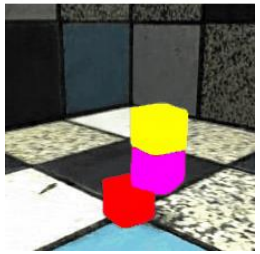
- Video prediction:
  - From a single image, generate a video
  - A prediction of future states
- Issues:
  - Handling multiple objects in frame
  - Multi-modality





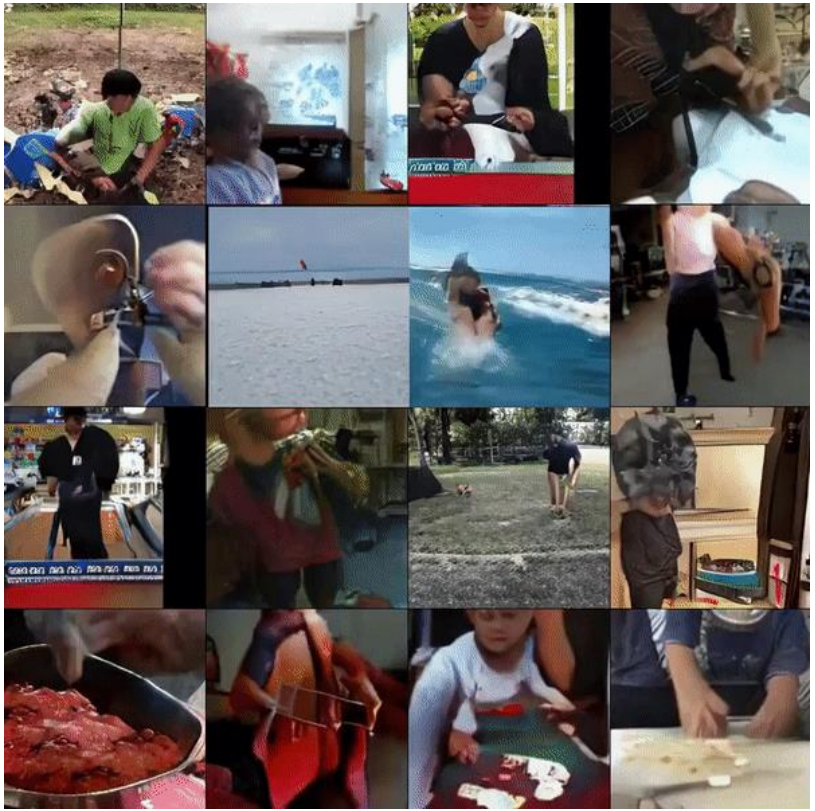






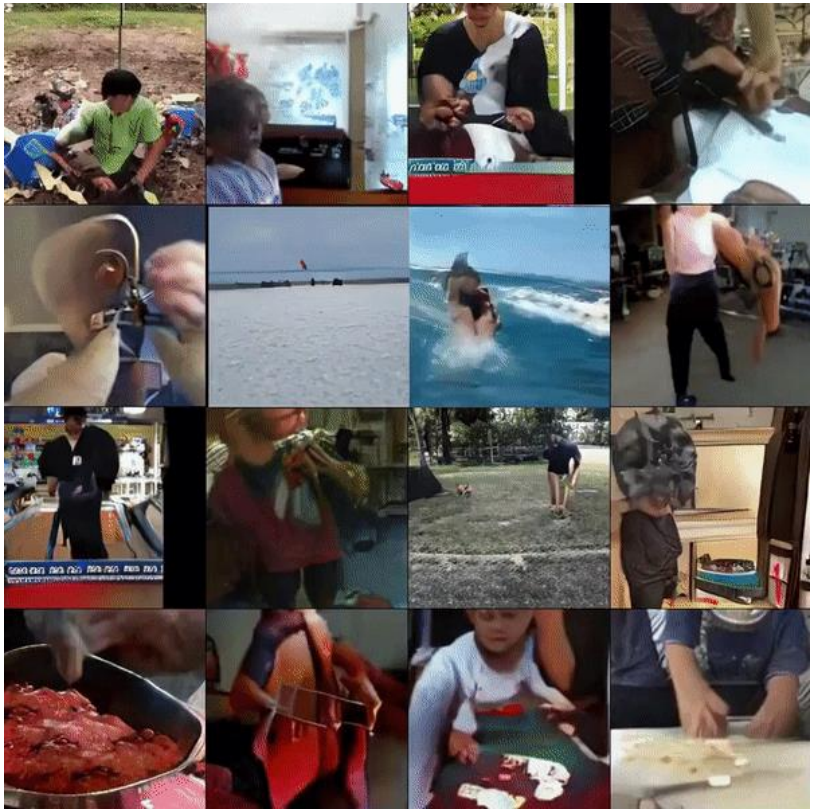
Related Works:
DVD-GAN

- Use noise and class-conditioning



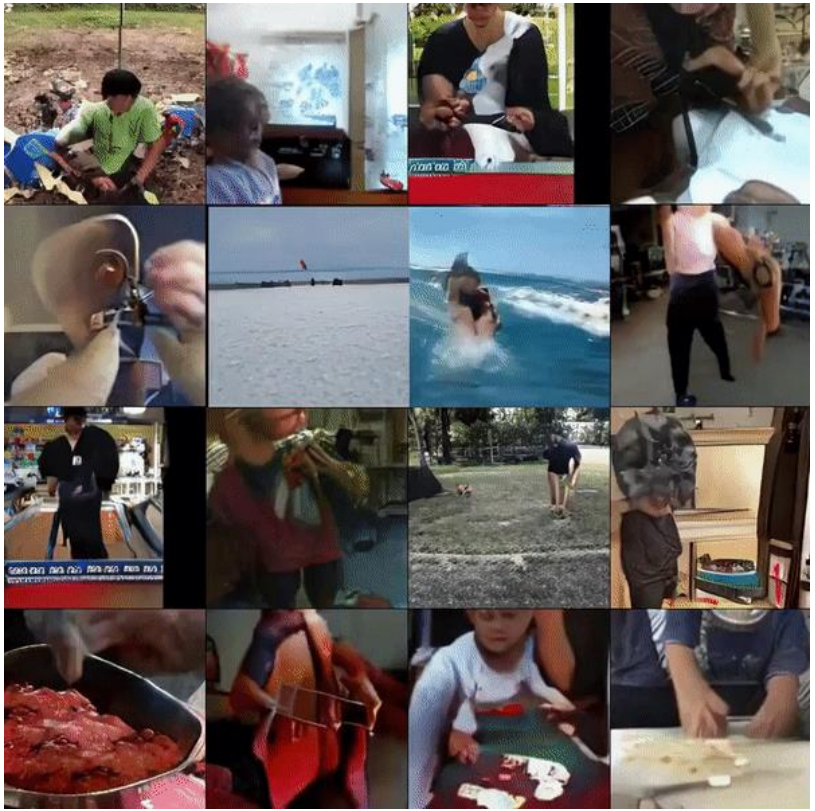
Related Works:
DVD-GAN

- Use noise and class-conditioning
- Generalized approach



Related Works:
DVD-GAN

- Use noise and class-conditioning
- Generalized approach
- Our problem is a subset



Related Works:

Monkey-Net

- Motion transfer using template matching



Related Works:

Monkey-Net

- Motion transfer using template matching
- Limited to one object



Related Works:

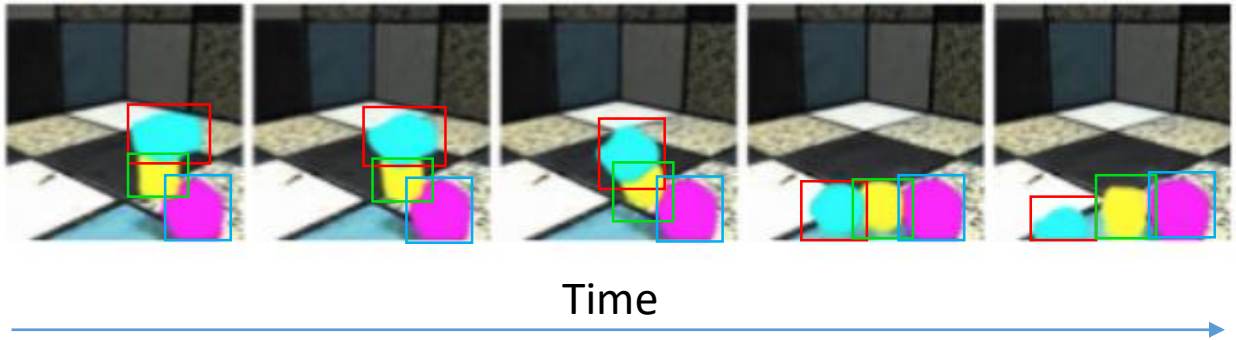
Monkey-Net

- Motion transfer using template matching
- Limited to one object
- Ours can handle multiple objects



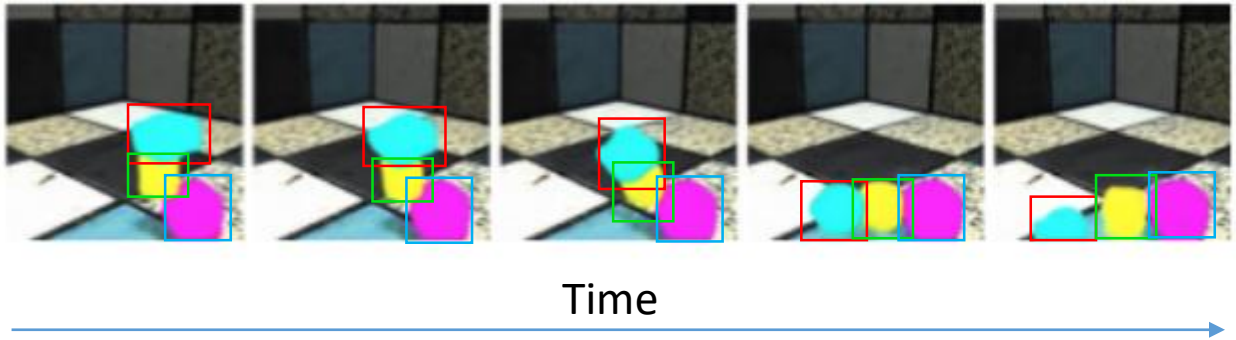
Proposed Method

- Avoid predicting on entire frame



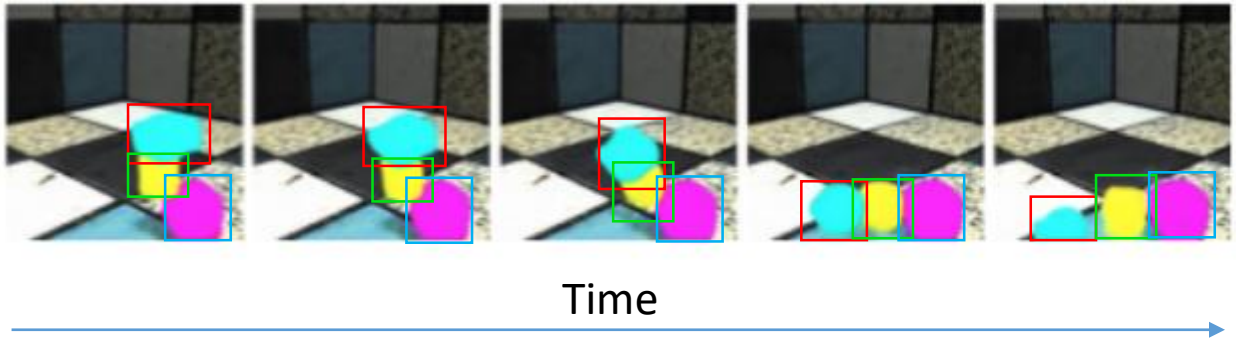
Proposed Method

- Avoid predicting on entire frame
- Model movements of “entities”



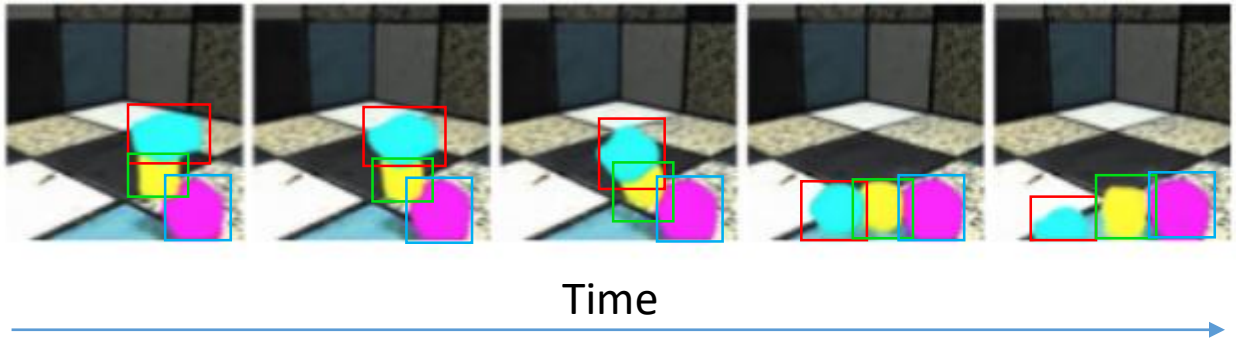
Proposed Method

- Avoid predicting on entire frame
- Model movements of “entities”
- Predictions are formed based on movement



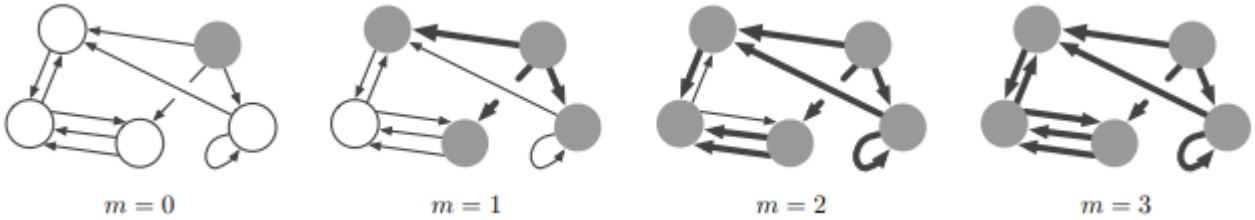
Proposed Method

- Avoid predicting on entire frame
- Model movements of “entities”
- Predictions are formed based on movement
- “Pixel-level” predictions



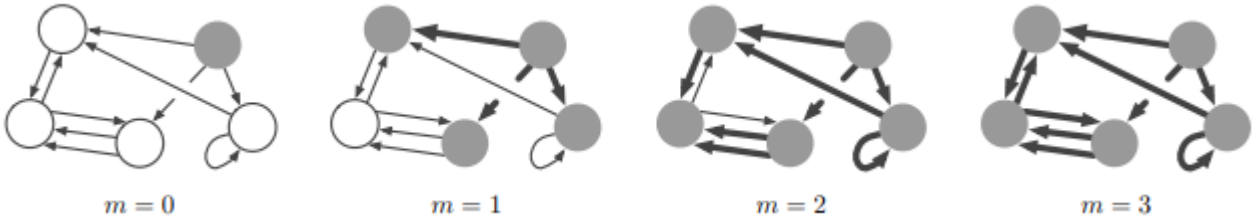
Graph Networks

- Units are multigraphs



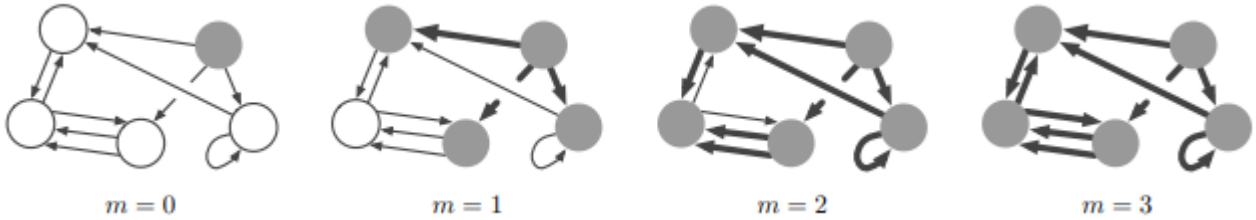
Graph Networks

- Units are multigraphs
- Nodes store features



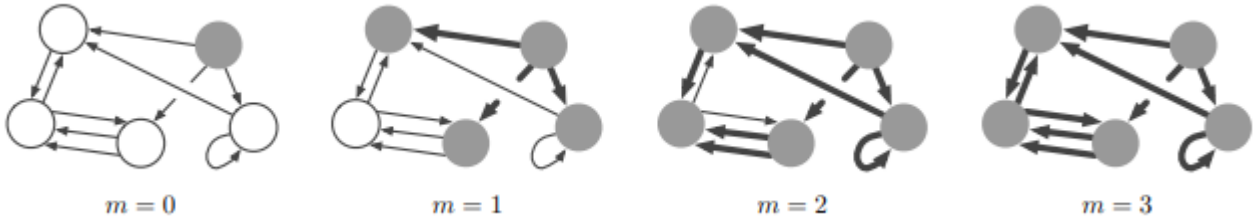
Graph Networks

- Units are multigraphs
- Nodes store features
- Edges form relationships between features



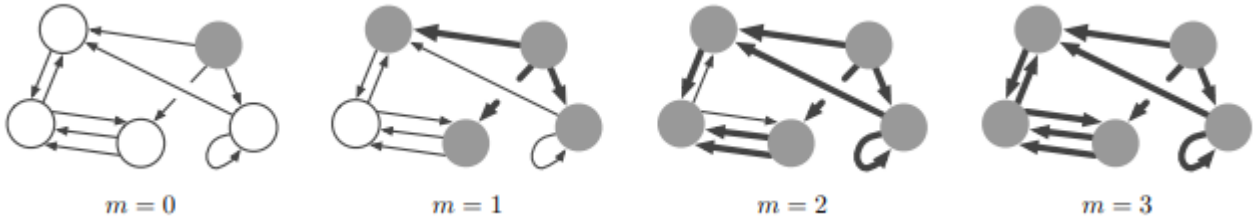
Graph Networks

- Units are multigraphs
- Nodes store features
- Edges form relationships between features
- Information diffused between nodes

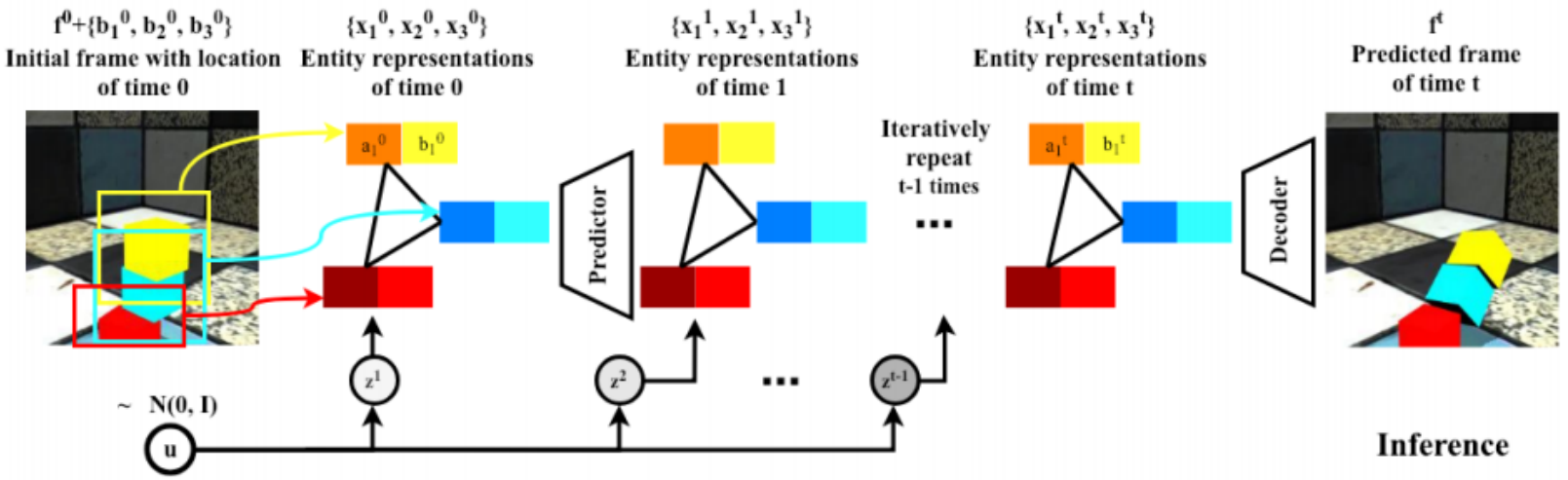


Graph Networks

- Units are multigraphs
- Nodes store features
- Edges form relationships between features
- Information diffused between nodes
- Iterative “message-passing”



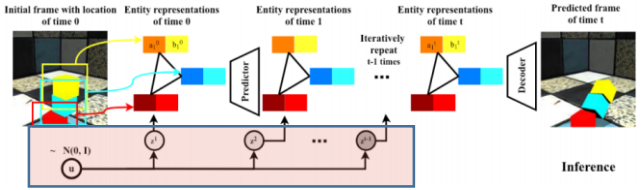
Proposed Method

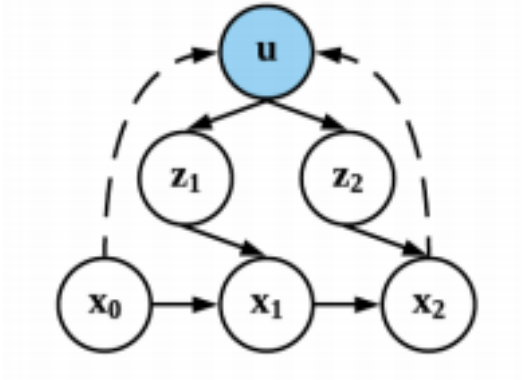


Proposed Method:

Latent Encoder

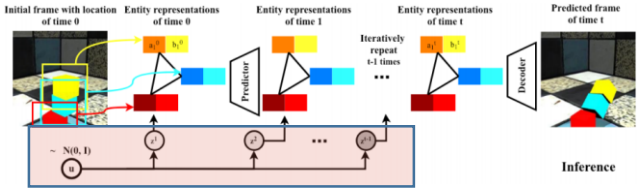
- Global latent variable *u*



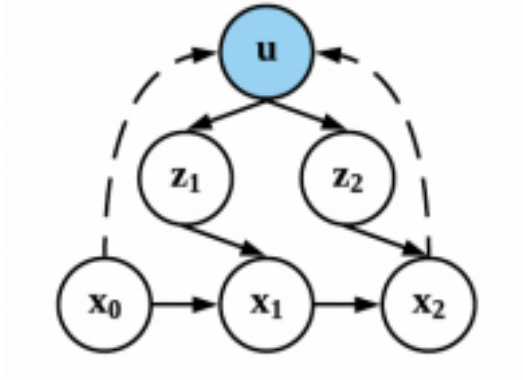


Proposed Method:

Latent Encoder

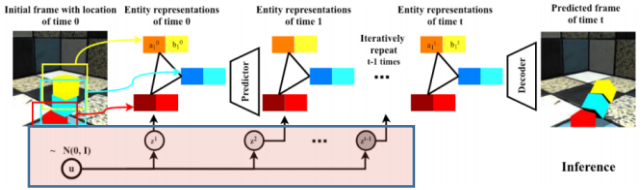


- Global latent variable *u*
- Remove ambiguity by learning motion

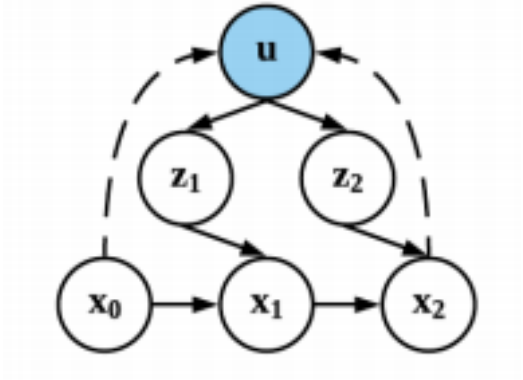


Proposed Method:

Latent Encoder

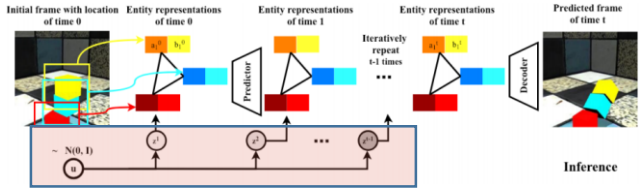


- Global latent variable *u*
- Remove ambiguity by learning motion
- Use a latent encoder to predict the distribution of *u*

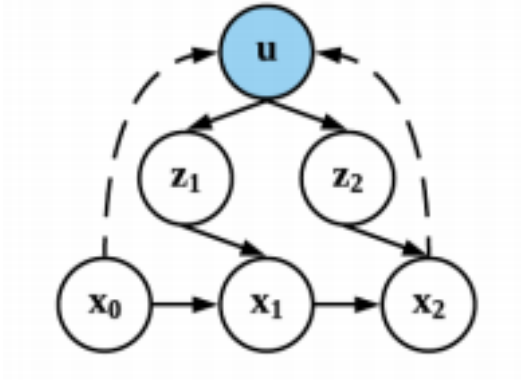


Proposed Method:

Latent Encoder

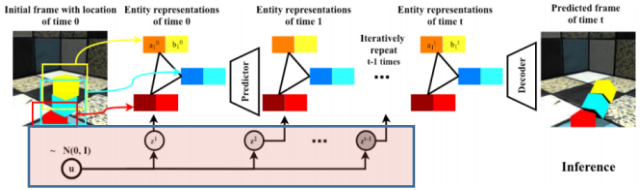


- Global latent variable *u*
- Remove ambiguity by learning motion
- Use a latent encoder to predict the distribution of *u*
- Obtained over all samples

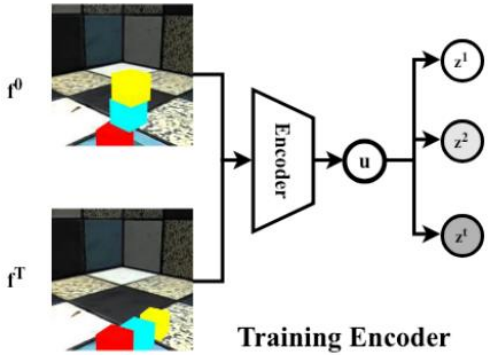


Proposed Method:

Latent Encoder

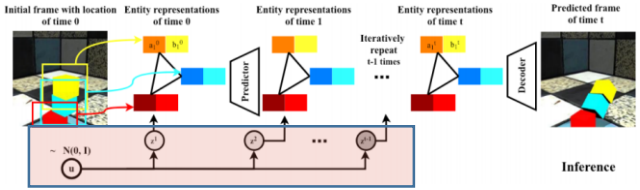


- Procedure:
  - Receive first (f_0) and last (f^T) frames of video

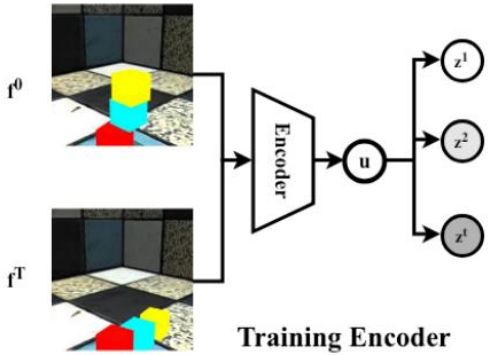


Proposed Method:

Latent Encoder

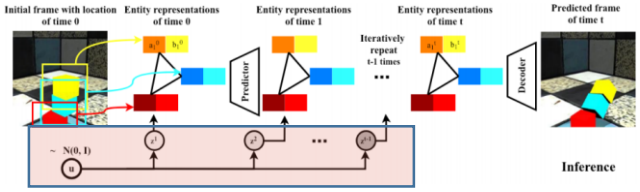


- Procedure:
  - Receive first (f_0) and last (f^T) frames of video
  - Learn distribution from input using feed-forward network

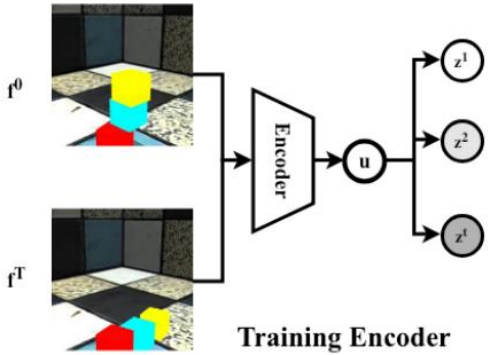


Proposed Method:

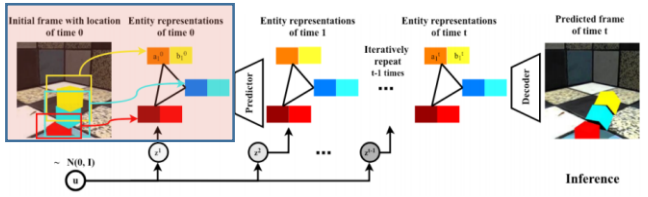
Latent Encoder



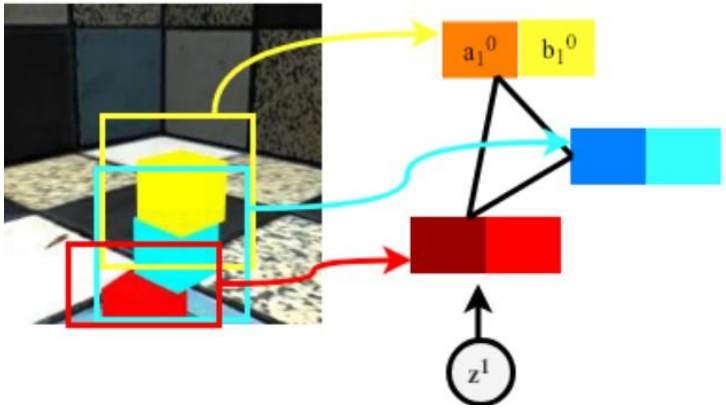
- Procedure:
  - Receive first (f_0) and last (f^T) frames of video
  - Learn distribution from input using feed-forward network
  - For any t, obtain z_t using one-layer LSTM on *u*



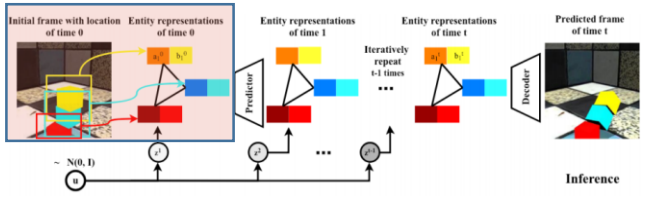
Proposed Method:



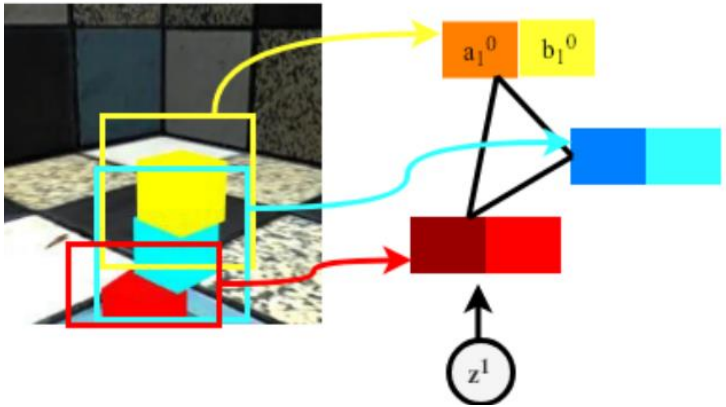
- Initial frame has annotated entities



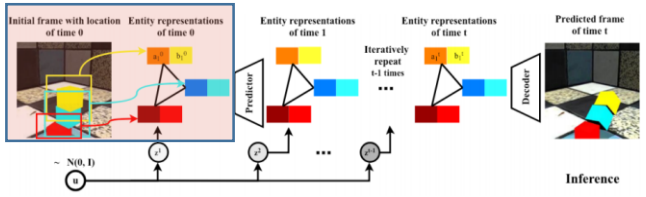
Proposed Method:



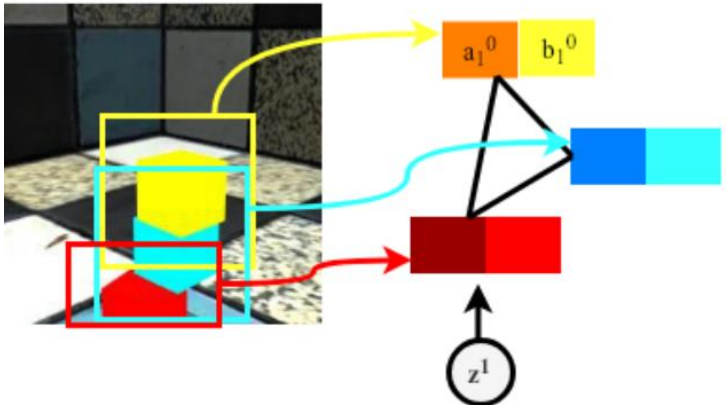
- Initial frame has annotated entities
- Features extracted via ResNet-18



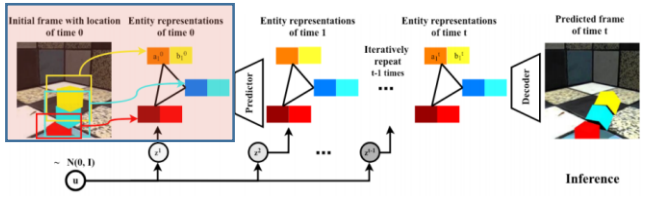
Proposed Method:



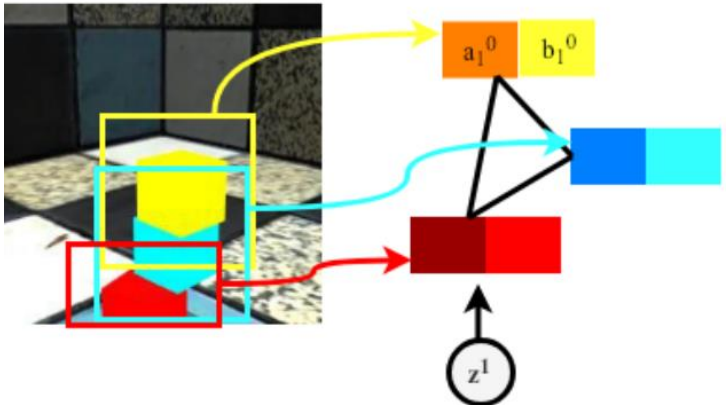
- Initial frame has annotated entities
- Features extracted via ResNet-18
- Features + annotations = entity representations



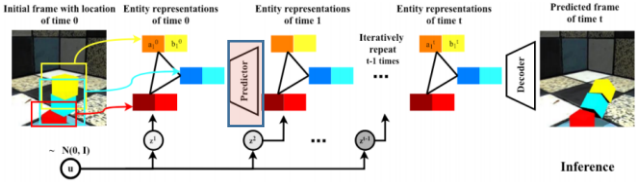
Proposed Method:



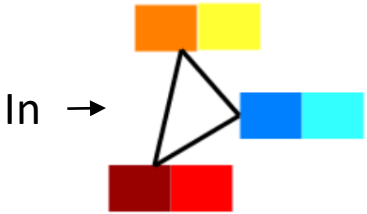
- Initial frame has annotated entities
- Features extracted via ResNet-18
- Features + annotations = entity representations
- Form graph with representations and latent feature



Proposed Method:
Entity Predictor

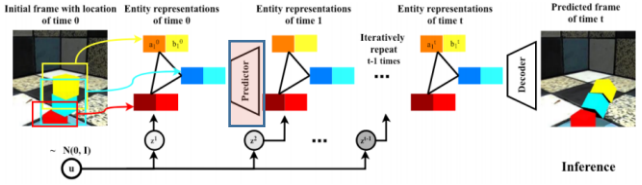


- Receive graph as input

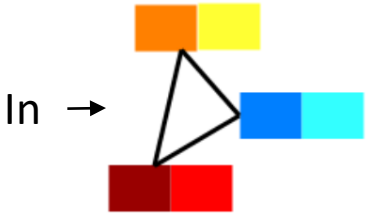


Proposed Method:

Entity Predictor

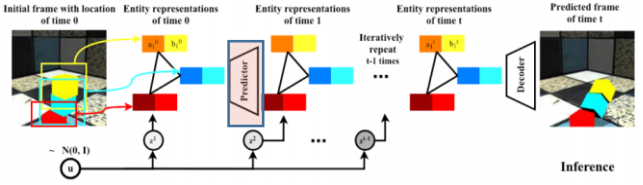


- Receive graph as input
- Predictor consists of 4 graph units

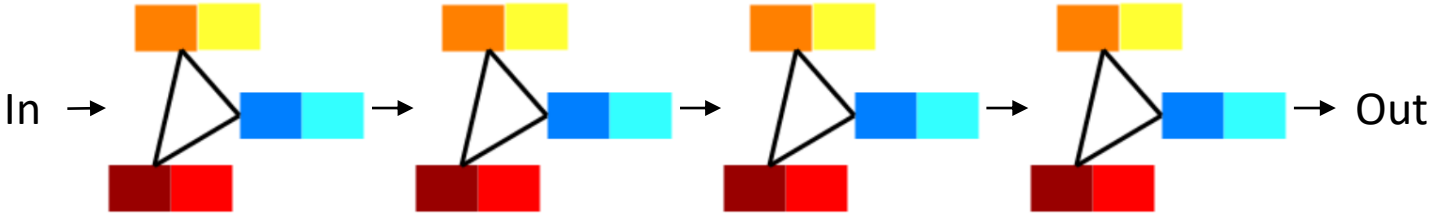


Proposed Method:

Entity Predictor

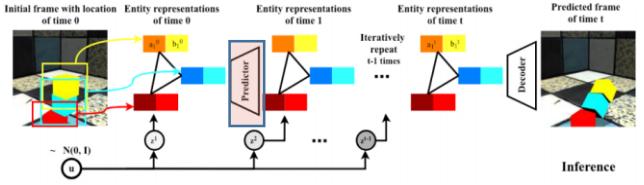


- Receive graph as input
- Predictor consists of 4 graph units

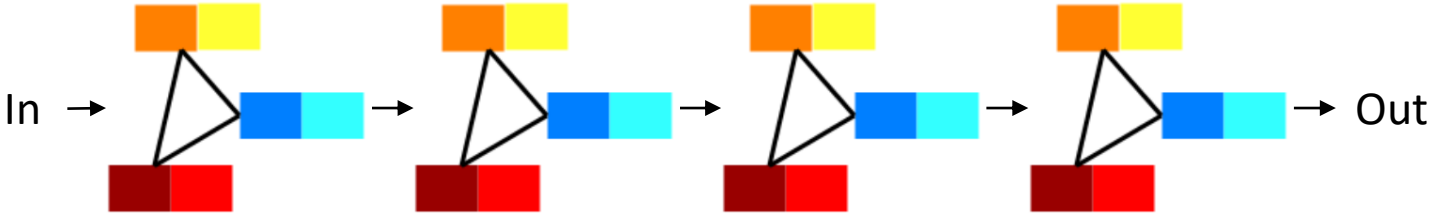


Proposed Method:

Entity Predictor

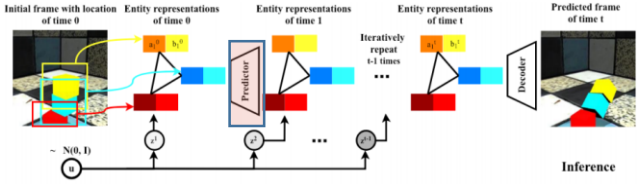


- Receive graph as input
- Predictor consists of 4 graph units
- Predictor infers changes in entities

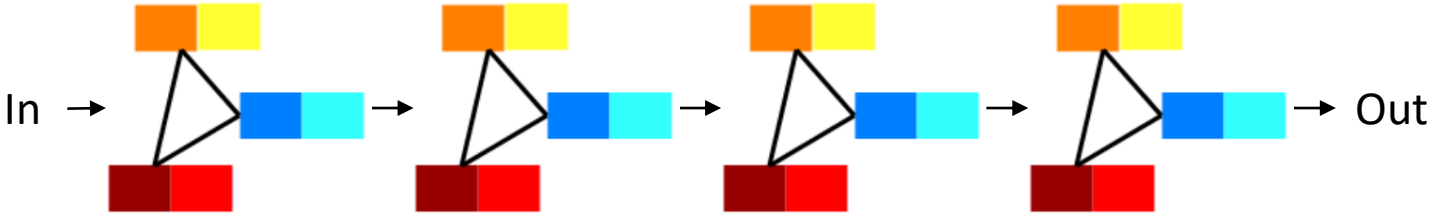


Proposed Method:

Entity Predictor

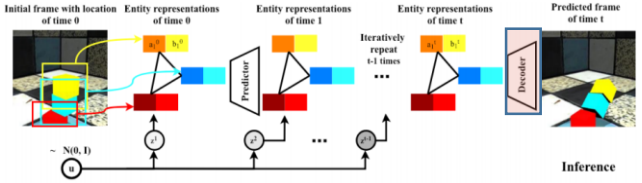


- Receive graph as input
- Predictor consists of 4 graph units
- Predictor infers changes in entities
- Allows for interactions between entities



Proposed Method:

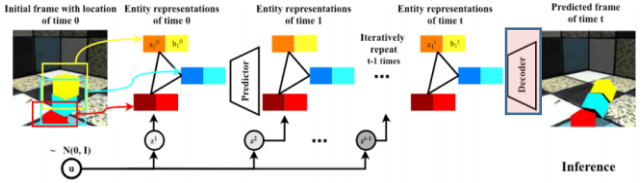
Frame Decoder



- Draw the frame from the representation

Proposed Method:

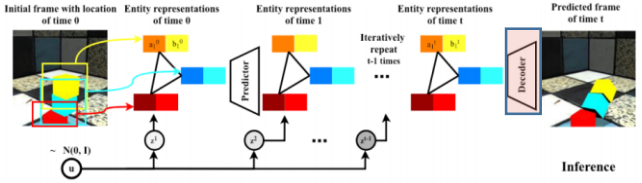
Frame Decoder



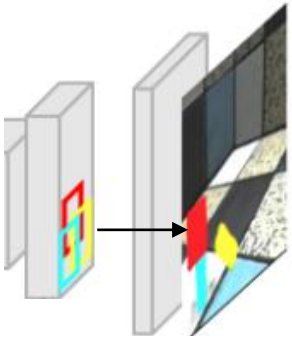
- Draw the frame from the representation
- Considerations:

Proposed Method:

Frame Decoder

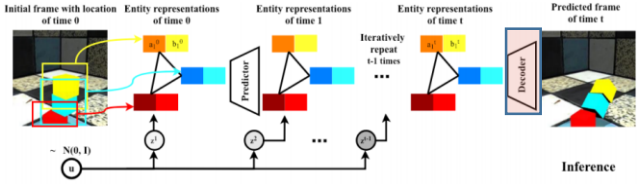


- Draw the frame from the representation
- Considerations:
  - Respect location of prediction

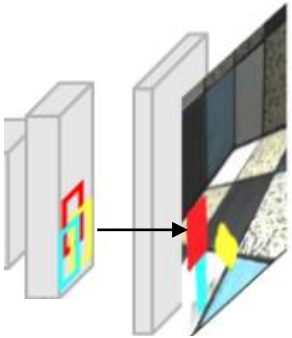


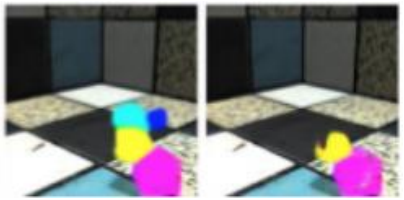
Proposed Method:

Frame Decoder



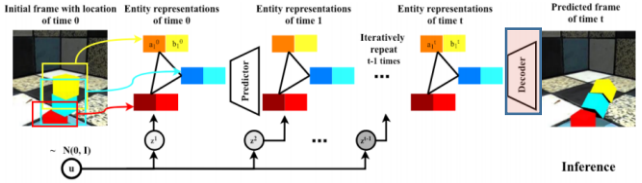
- Draw the frame from the representation
- Considerations:
  - Respect location of prediction
  - Handle occlusion and entity merging



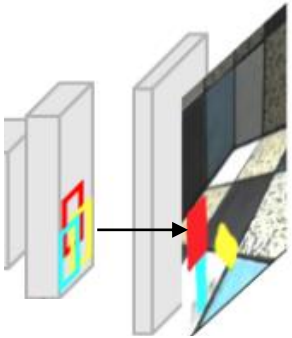


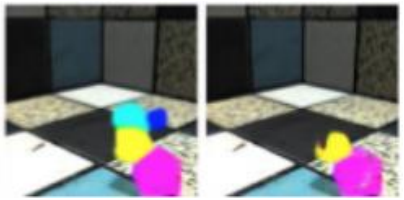
Proposed Method:

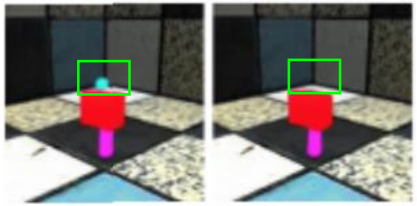
Frame Decoder



- Draw the frame from the representation
- Considerations:
  - Respect location of prediction
  - Handle occlusion and entity merging
  - New background may be visible

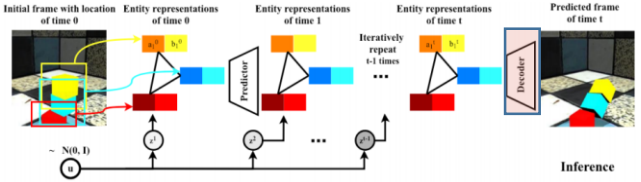






Proposed Method:

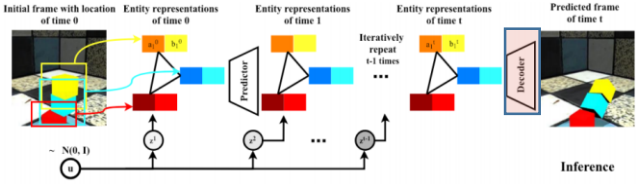
Frame Decoder



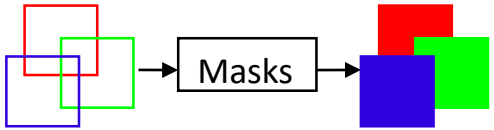
- Decode the per-entity spatial representation

Proposed Method:

Frame Decoder

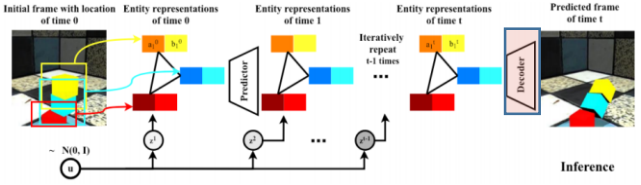


- Decode the per-entity spatial representation
  - Predict masks that determine visibility

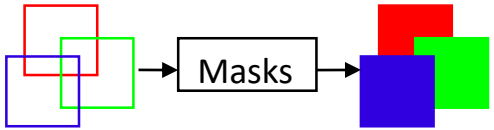


Proposed Method:

Frame Decoder

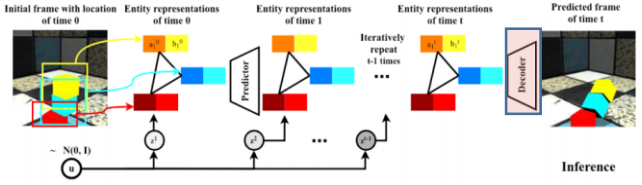


- Decode the per-entity spatial representation
  - Predict masks that determine visibility
  - Warp the features and masks to coordinates

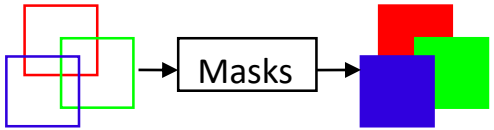


Proposed Method:

Frame Decoder

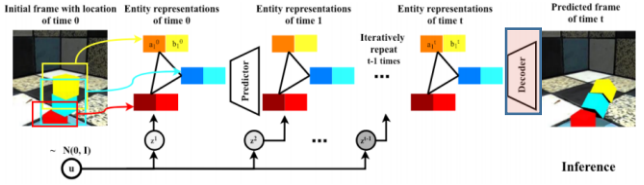


- Decode the per-entity spatial representation
  - Predict masks that determine visibility
  - Warp the features and masks to coordinates
- Draw new frame via pixel predictions using:

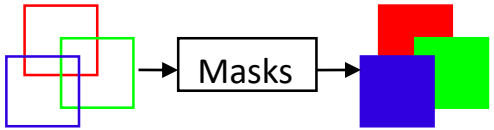


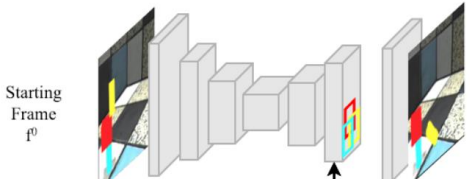
Proposed Method:

Frame Decoder



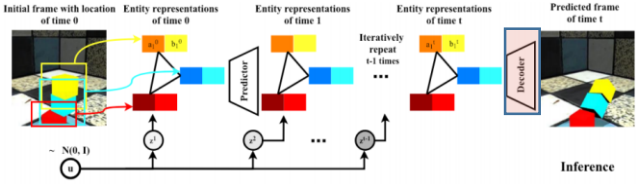
- Decode the per-entity spatial representation
  - Predict masks that determine visibility
  - Warp the features and masks to coordinates
- Draw new frame via pixel predictions using:
  - Features from starting frame, f_0



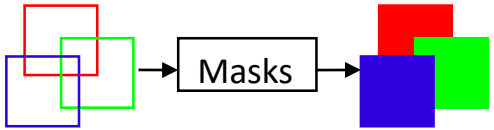


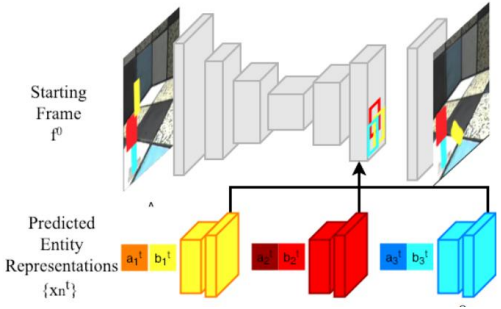
Proposed Method:

Frame Decoder

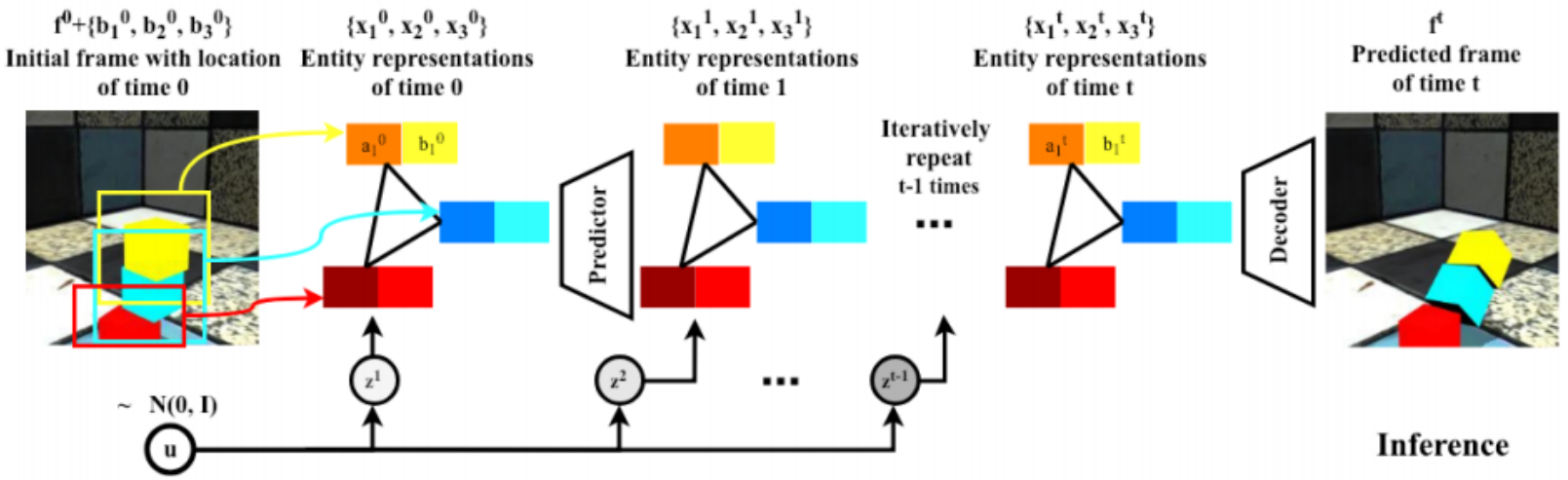


- Decode the per-entity spatial representation
  - Predict masks that determine visibility
  - Warp the features and masks to coordinates
- Draw new frame via pixel predictions using:
  - Features from starting frame, f_0
  - Predicted coordinates





Proposed Method



Proposed Method:

Objective Function / Loss

- Loss from predicted frame / locations to ground truth:

Proposed Method:

Objective Function / Loss

- Loss from predicted frame / locations to ground truth:

$$L_{pred} = \sum_{t=1}^{T}(\|\mathcal{D}(\{x_n^t\}, f^0) - \hat{f}^t\|_1 + \lambda_1 \sum_{n=1}^{N} \|b_n^t - \hat{b}_n^t\|^2)$$

Proposed Method:

Objective Function / Loss

- Loss from predicted frame / locations to ground truth:

$$L_{pred} = \sum_{t=1}^{T}(\|\mathcal{D}(\{x_n^t\}, f^0) - \hat{f}^t\|_1 + \lambda_1 \sum_{n=1}^{N} \|b_n^t - \hat{b}_n^t\|^2)$$

Frame loss

Hyperparameter

Location loss

Proposed Method:

Objective Function / Loss

- Loss from bottleneck on latent variable u:

Proposed Method:

Objective Function / Loss

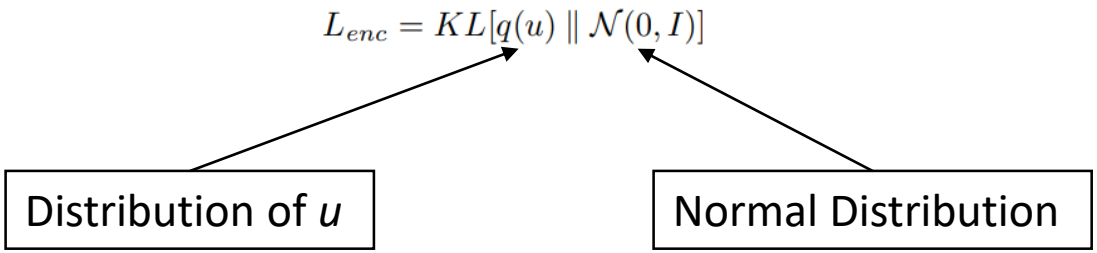
- Loss from bottleneck on latent variable u:

$$L_{enc} = KL[q(u) \parallel \mathcal{N}(0, I)]$$

Proposed Method:

Objective Function / Loss

- Loss from bottleneck on latent variable u:



Proposed Method:

Objective Function / Loss

- Auto-encoding loss for the decoder

Proposed Method:

Objective Function / Loss

- Auto-encoding loss for the decoder

$$L_{dec} = \sum_{t=0}^{T} \|\mathcal{D}(\{\hat{x}_n^t\}, f^0) - \hat{f}^t\|_1$$

Proposed Method:

Objective Function / Loss

- Auto-encoding loss for the decoder

$$L_{dec} = \sum_{t=0}^{T} \|\mathcal{D}(\{\hat{x}_n^t\}, f^0) - \hat{f}^t\|_1$$

Entity representations from GT

Proposed Method:

Objective Function / Loss

- Loss from predicted frame / locations to ground truth:

$$L_{pred} = \sum_{t=1}^{T}(\|\mathcal{D}(\{x_n^t\}, f^0) - \hat{f}^t\|_1 + \lambda_1 \sum_{n=1}^{N} \|b_n^t - \hat{b}_n^t\|^2)$$

- Loss from bottleneck on latent variable u:

$$L_{enc} = KL[q(u) \parallel \mathcal{N}(0, I)]$$

- Auto-encoder loss for the decoder:

$$L_{dec} = \sum_{t=0}^{T} \|\mathcal{D}(\{\hat{x}_n^t\}, f^0) - \hat{f}^t\|_1$$

- Overall Loss:

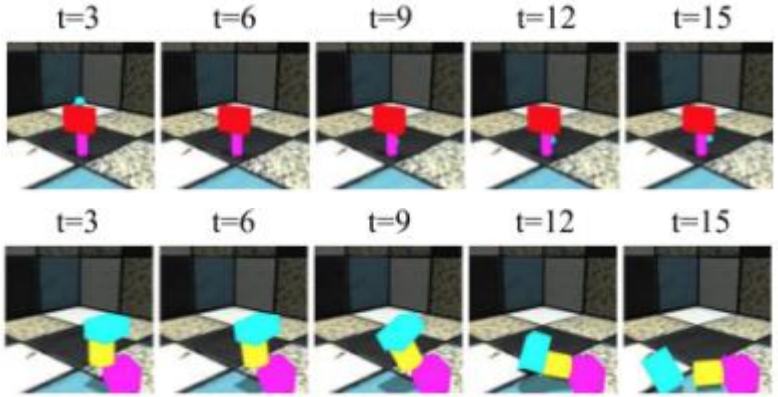
$$L = L_{dec} + L_{pred} + \lambda_2 L_{enc}$$

- λ_1, λ_2 are hyperparameters

Experiments:

ShapeStacks

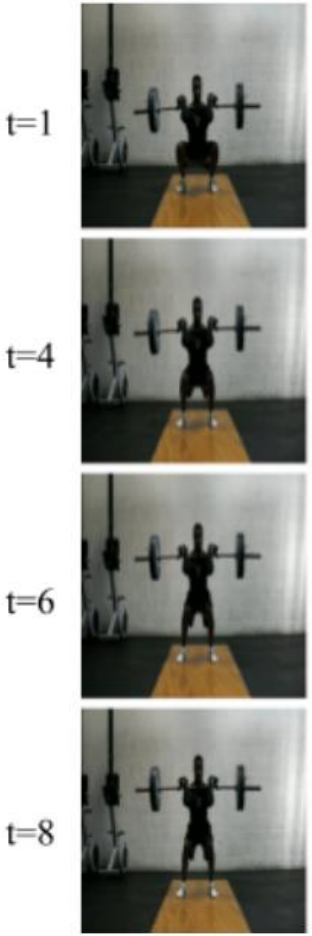
- Synthetic video dataset consisting of stacked “blocks”
  - Blocks fall due to gravity
  - “Blocks”: cubes, cylinders, or spheres
- To evaluate generalization ability, videos with 3 – 6 blocks used



Experiments:

Penn Action

- Real dataset consisting of human activity
  - Indoor and outdoor sports with annotations of human joints
- Use subset of videos relating to gym activities
  - No camera motion
  - Backgrounds are similar

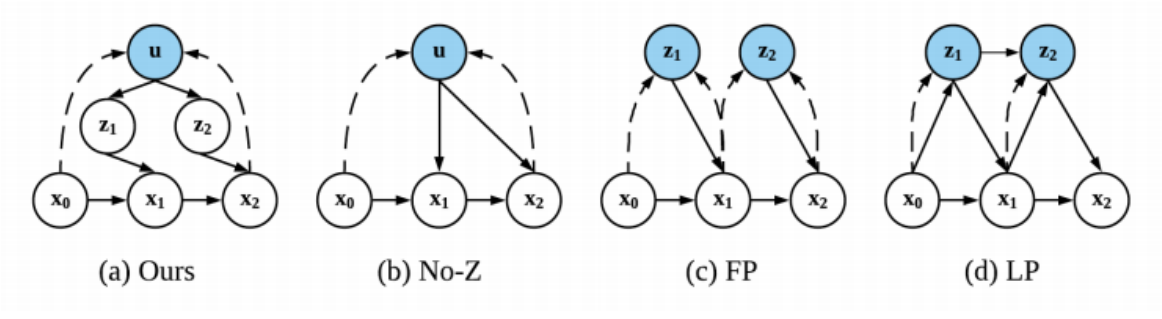


Experiments:

Baselines

- Baselines used:
  - No-Factor (predicts frames directly)
  - No-Edge (does not allow for entity interaction)
  - Pose Knows (implicit location prediction)

- Alternative prediction policies:
  - Changes to the encoder module



Experiments:

Evaluation Metrics

- Average Mean Square Error (MSE) for location accuracy
- Predicted entity locations are compared to ground truth

$$MSE = \frac{1}{t}\sum_{i=1}^{t}(X - \hat{X})^2$$

Experiments:

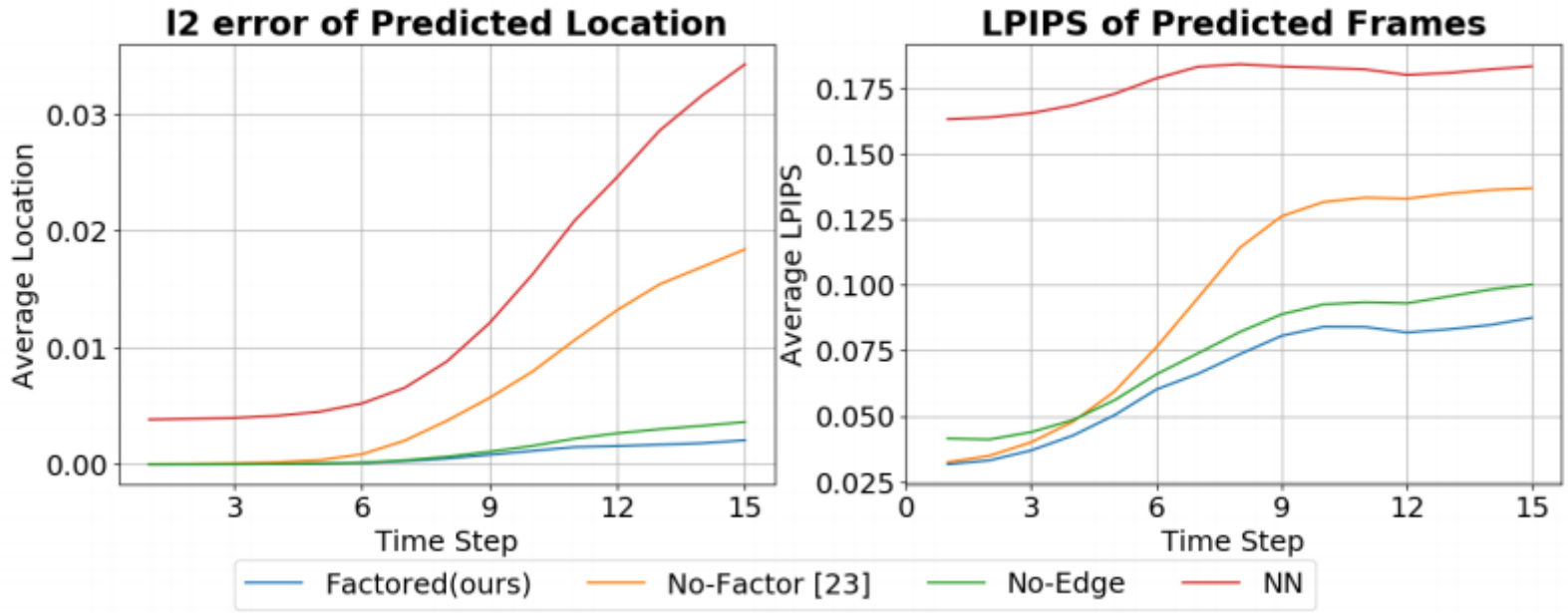
Evaluation Metrics

- Learned Perceptual Image Patch Similarity (LPIPS) for quality
- Compares the similarity of features of two images
- Unit-normalized comparison across all channels

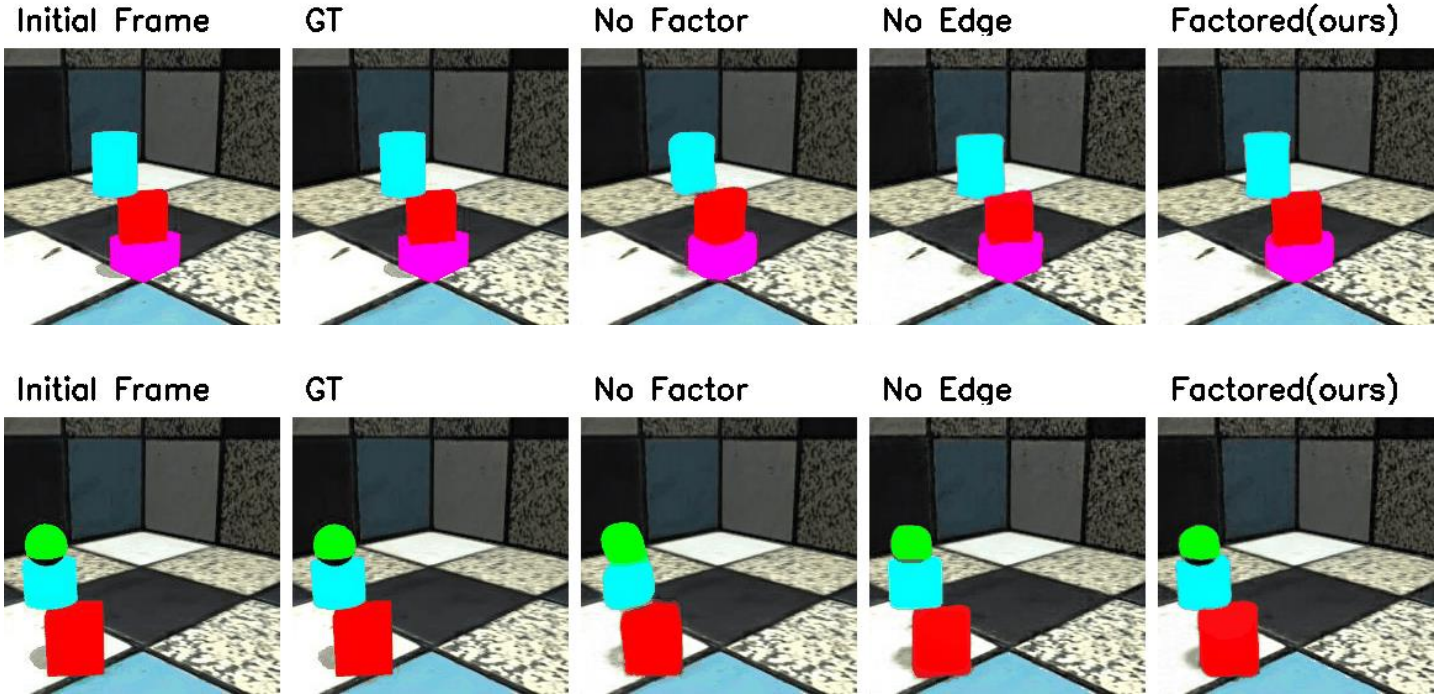
$$d(x, x_0) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} ||(y - y_0)||^2$$

Results:
ShapeStacks

- Comparison to Baseline

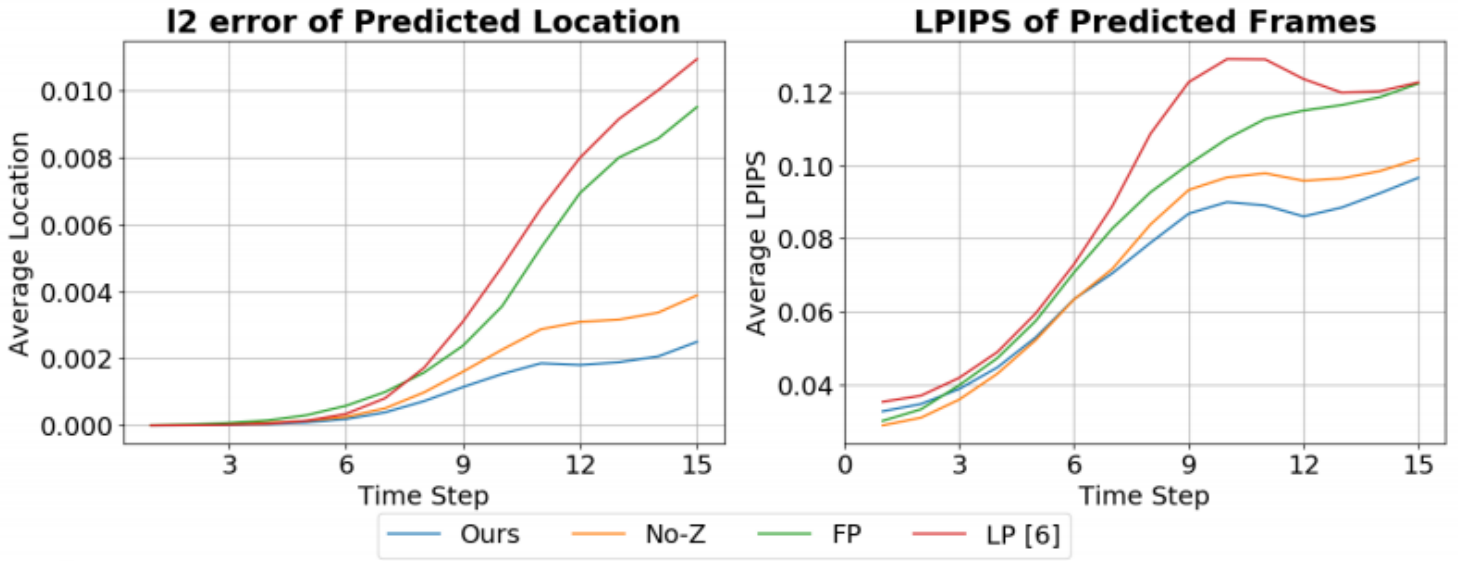


Results:
ShapeStacks

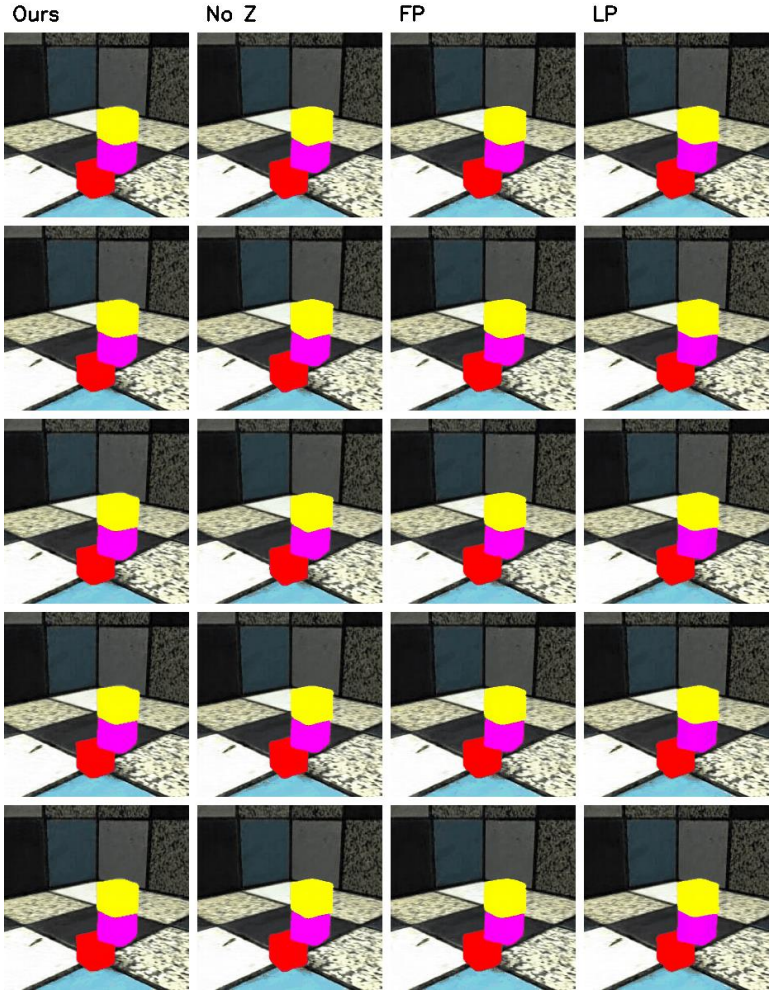


Results:
ShapeStacks

- Differing predicting policies

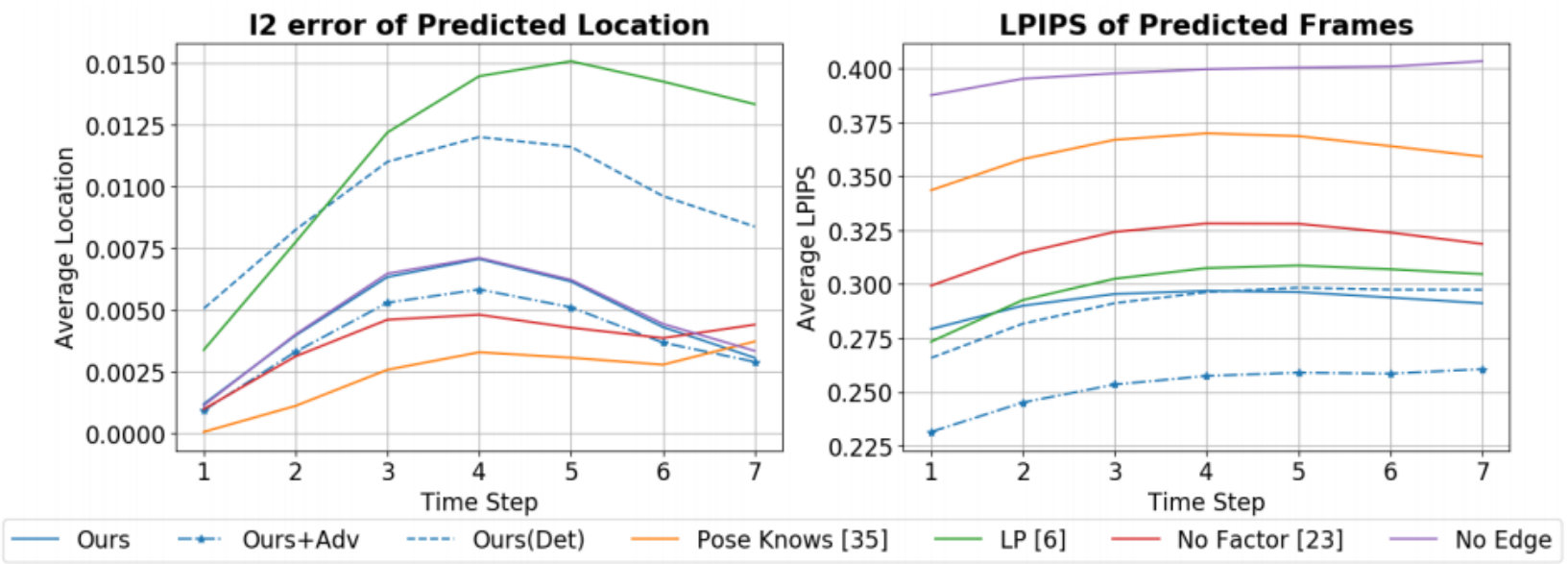


Results:
ShapeStacks



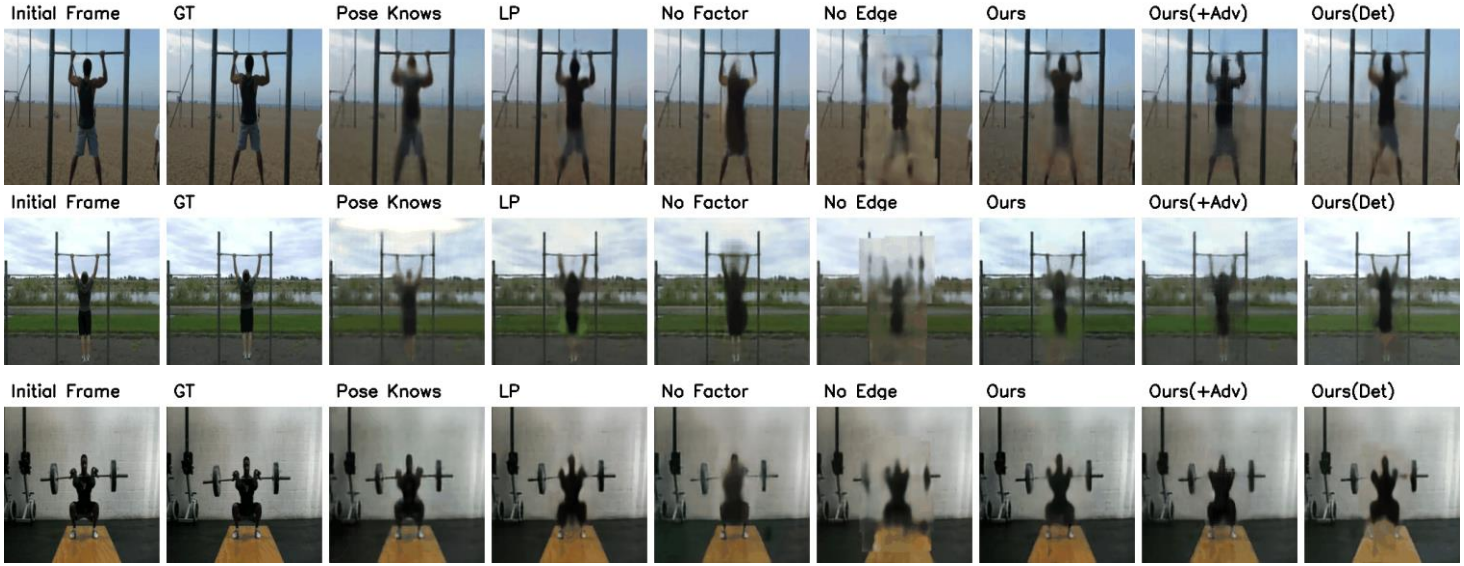
Results:

Penn Action



Results:

Penn Action




UNIVERSITY OF CENTRAL FLORIDA
1963
UCF