ellis | UNITRENTO

Cross-modal Video Generation

NICU SEBE
Univ. of Trento
niculae.sebe@unitn.it

ELLIS Summer school
Large-scale Artificial Intelligence
Modena, Italy 18-22 September 2023

Collaborators: Xavier Alameda-Pineda, Stephane Lathuiliere, Willi Menapace, Elisa Ricci, Subhankar Roy, Aliaksandr Siarohin, Hao Tang, Sergey Tulyakov, etc.

Deep Fakes: Driving Video, Static Input

Deep Fakes: Video/Voice Inpainting

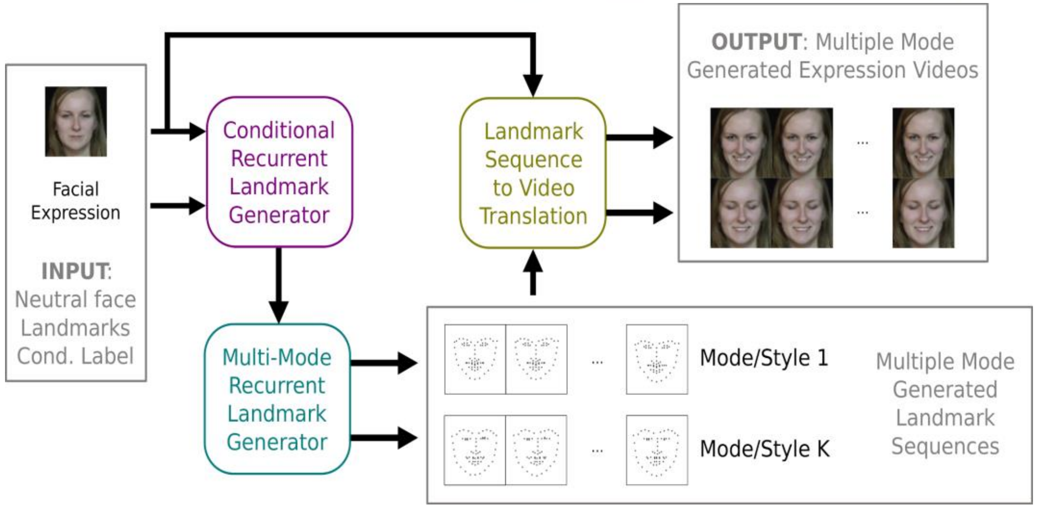
Creating Games with Real Footage

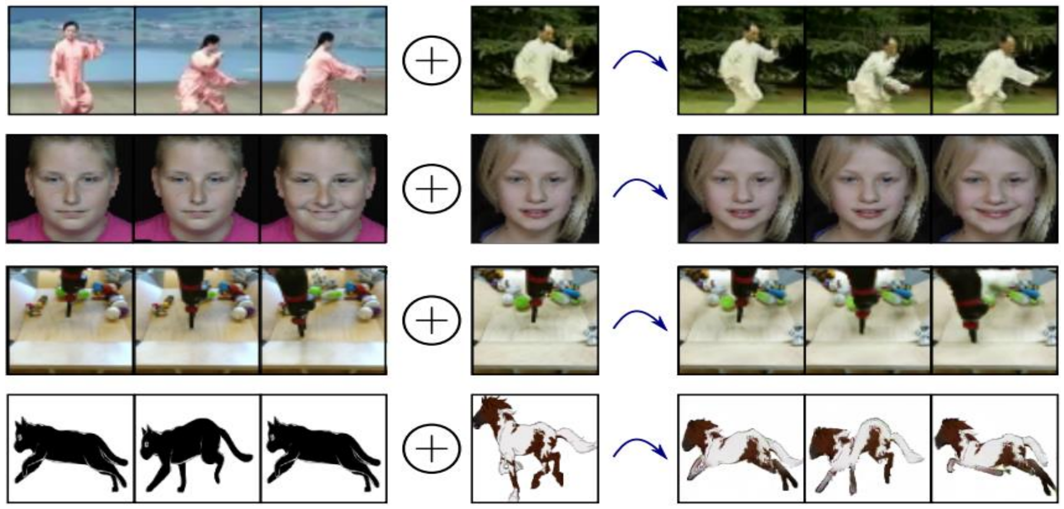
The player moves to the left corner waiting for the serve

The player serves the ball to the left corner of the field

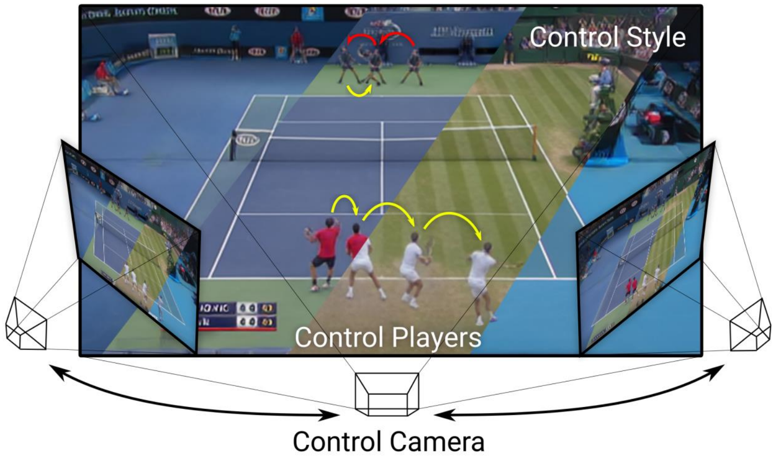
Image and Video Generation

Deep Generative Models for Image/Video Generation & Animation





Arbitrary Object Animation with/without 3D Modeling

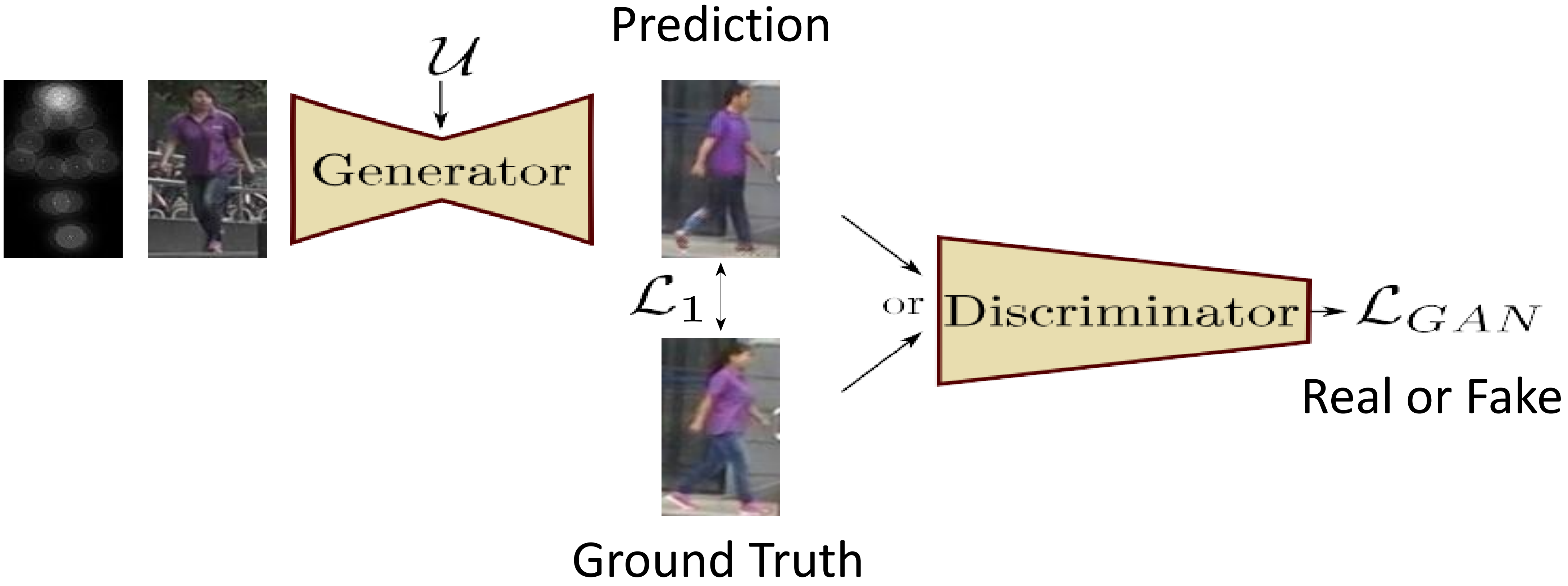


Pose-based Human Image Generation

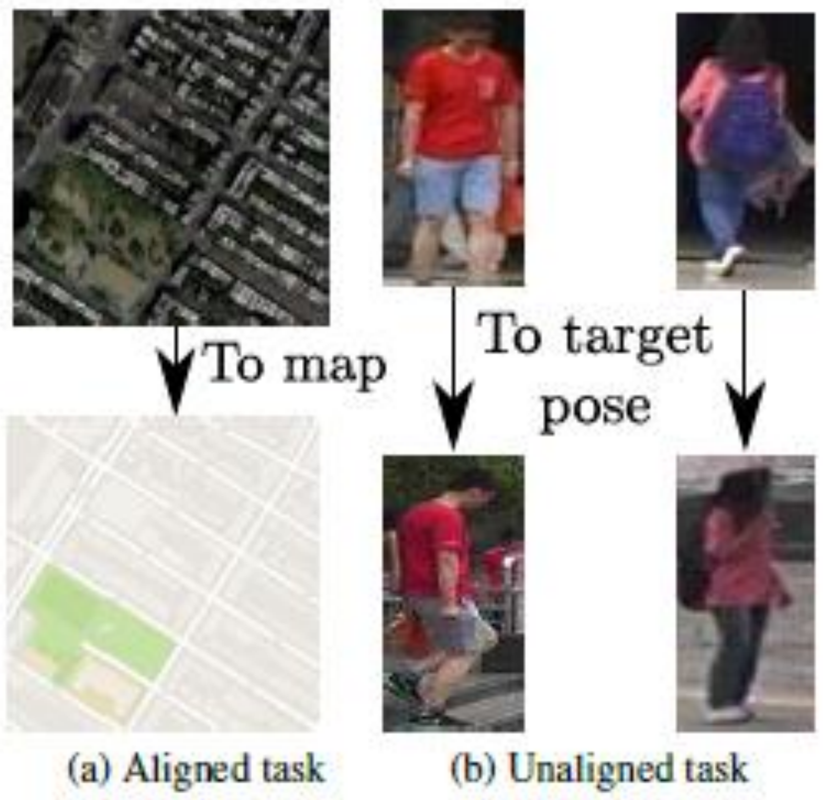
- Siarohin, et al., "Deformable GANs for Pose-based Human Image Generation", CVPR18
- Siarohin, et al., "Appearance and Pose-Conditioned Human Image Generation using Deformable GANs", PAMI, 43(4):1156-1171, April 2021

https://github.com/AliaksandrSiarohin/pose-gan

Pose-based Human Image Generation

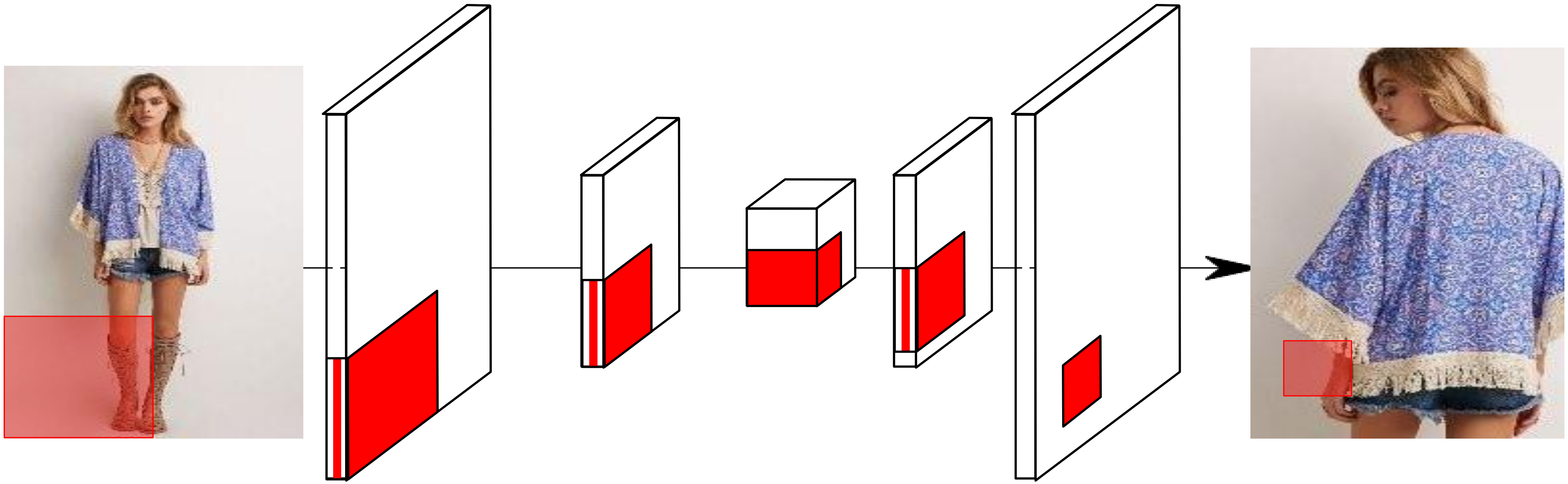


Pose-based Human Image Generation

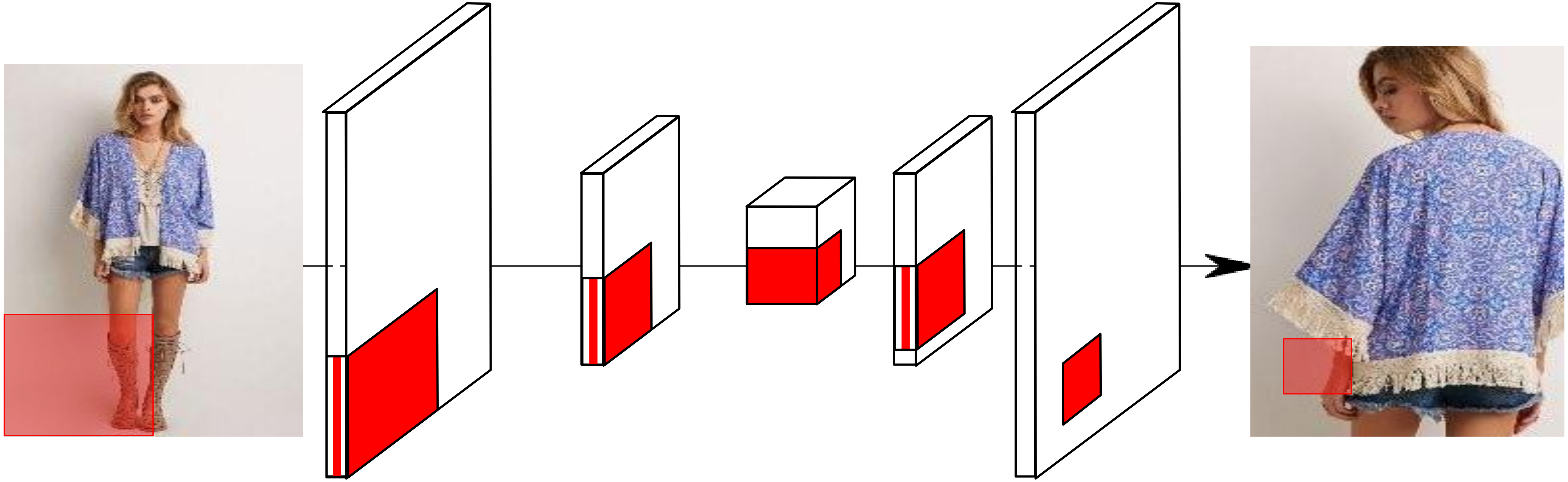


(a) typical "rigid" scene generation task: the local structures of conditioning and output image local structures are well aligned
(b) deformable-object generation task: the input and output are not spatially aligned

Pose-based Human Image Generation

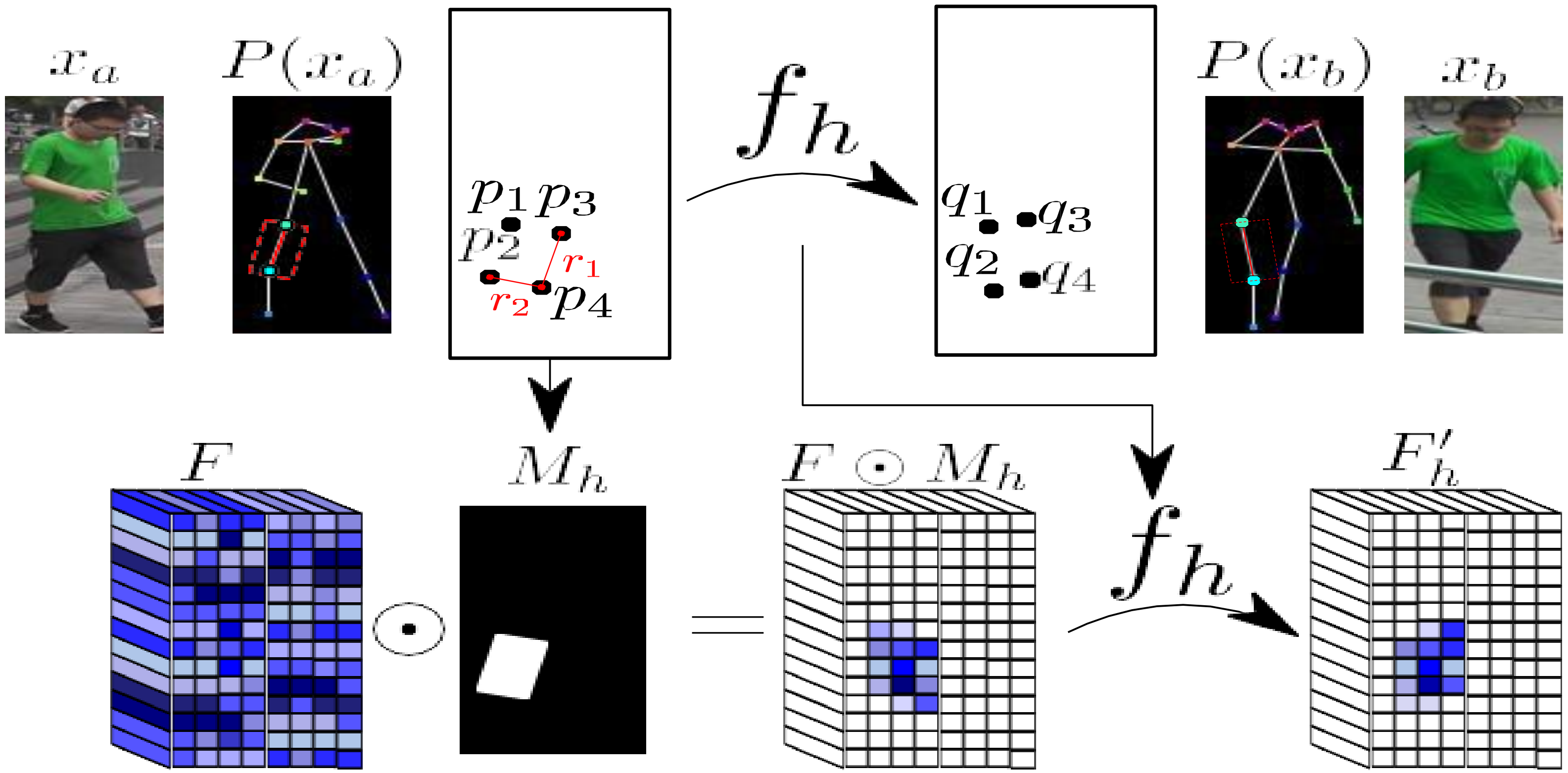


Pose-based Human Image Generation



We need a deformation model

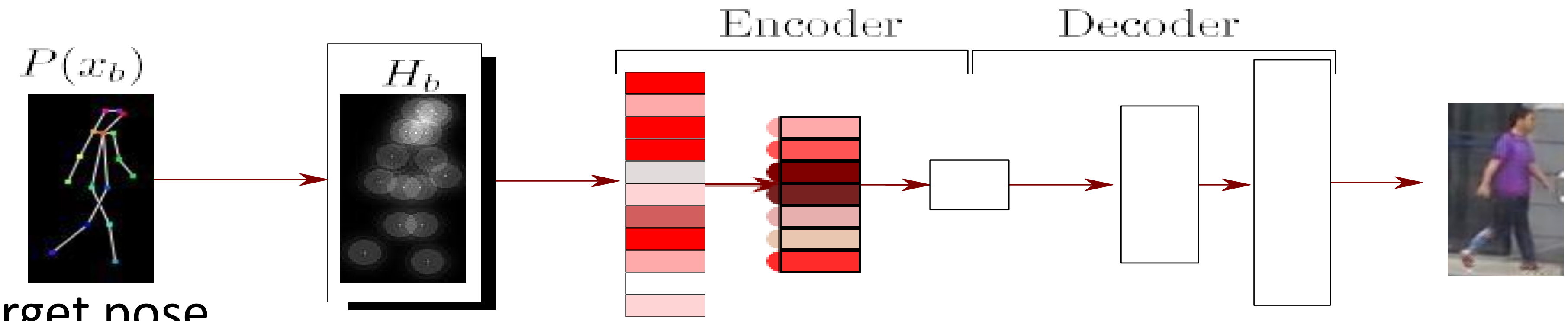
Pose-based Human Image Generation



- For each specific body part, compute an affine transformation f_h
- Use f_h to “move” the corresponding feature-map content

Pose-based Human Image Generation

Target Stream



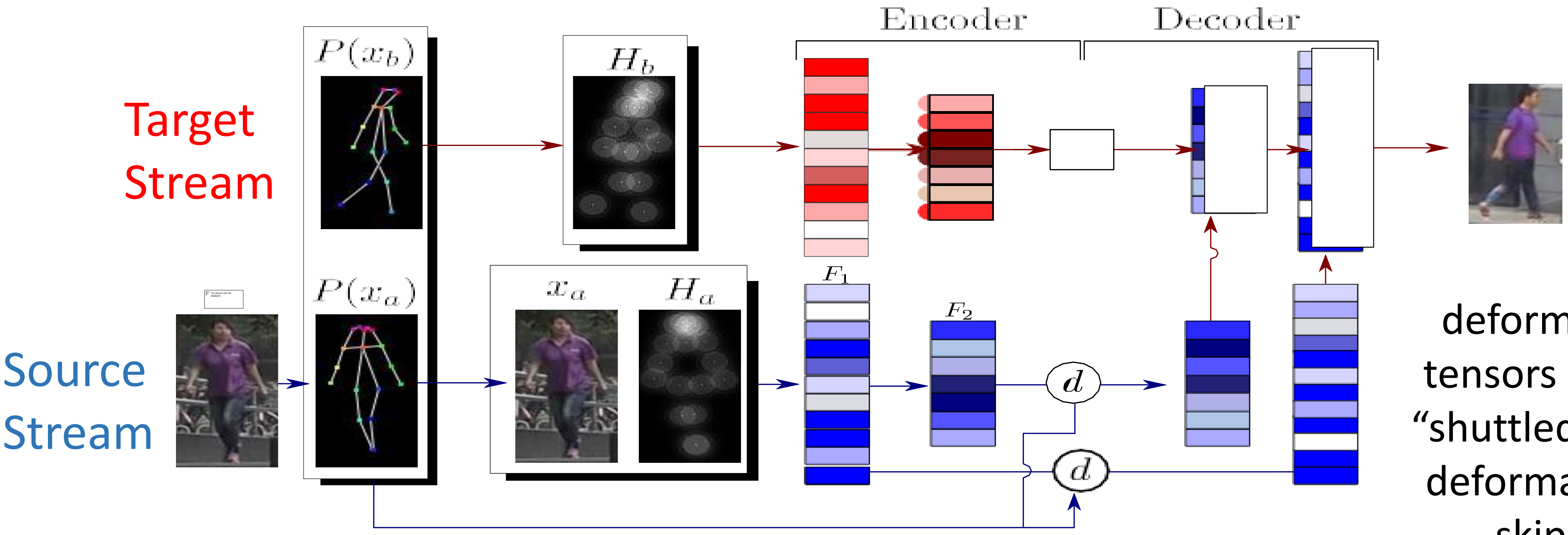
target pose

heat map using gaussian blurring

feature maps “shuttled” by skip connections from the H_b stream

feature maps directly obtained using up-convolutional filters applied to the previous-layer maps

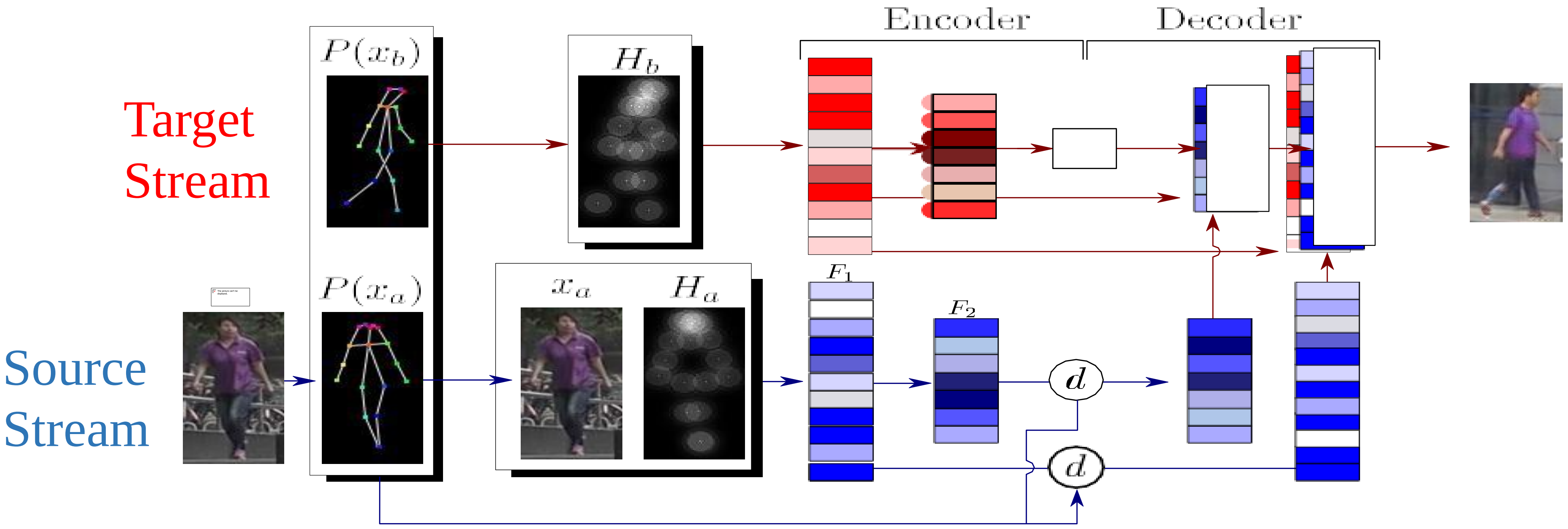
Pose-based Human Image Generation



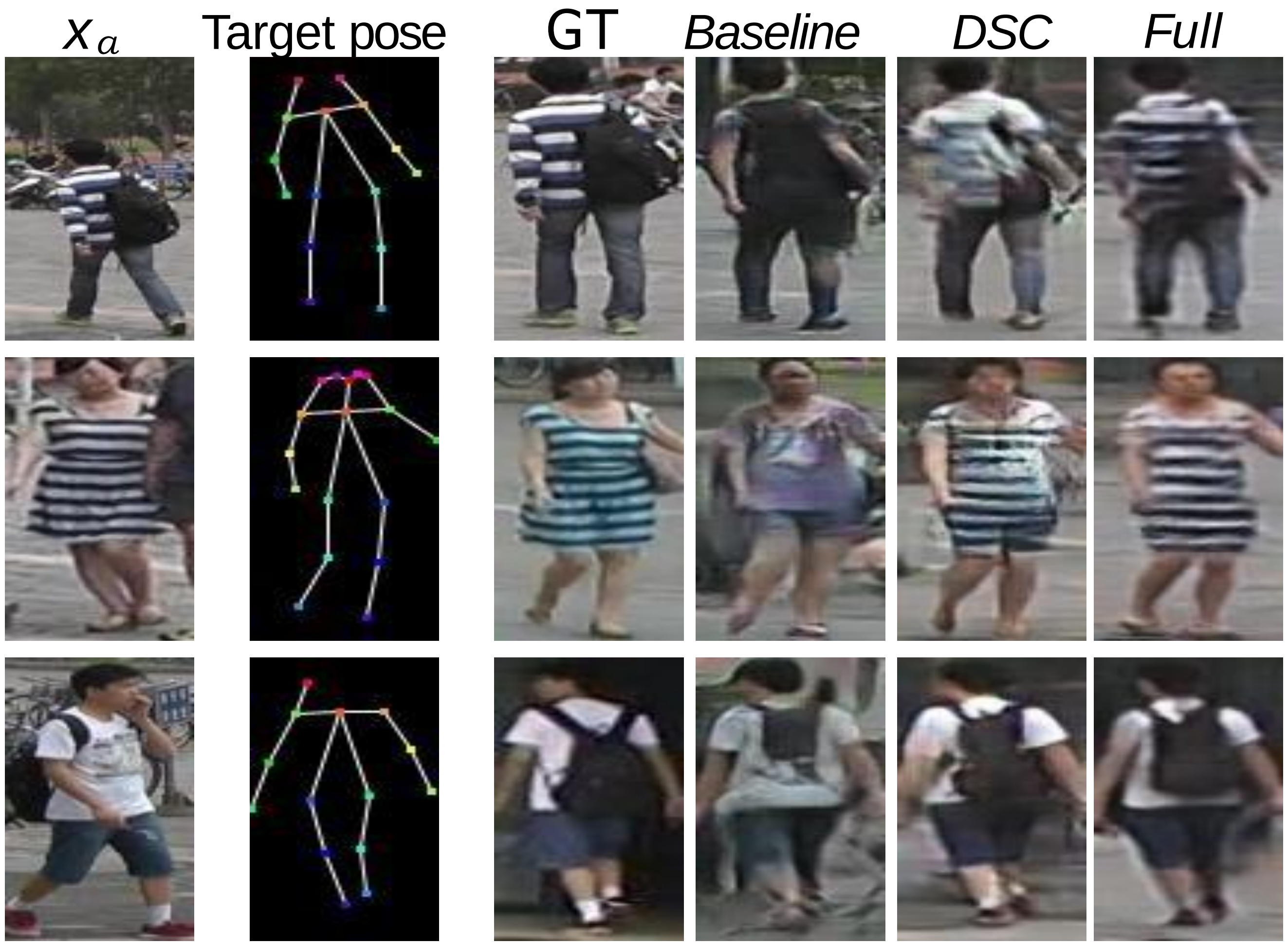
deformed tensors d(F) “shuttled” by deformable skip connections from (x_a,H_a) stream

- joint locations in x_a and H_a are spatially aligned (by construction)
- in H_b the joint locations may be far apart from x_a
- Hence, H_b is not concatenated with the other input tensors

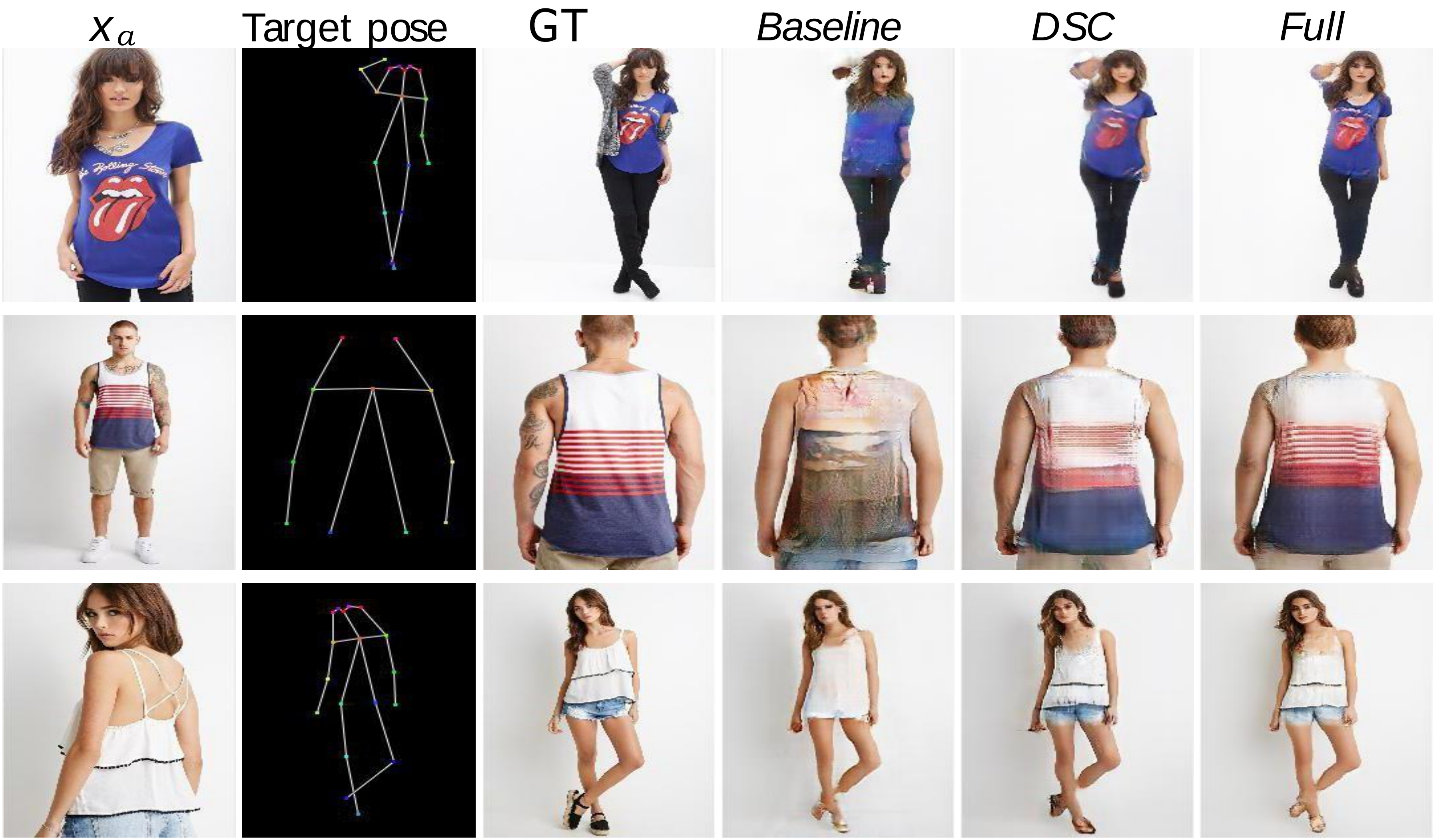
Pose-based Human Image Generation



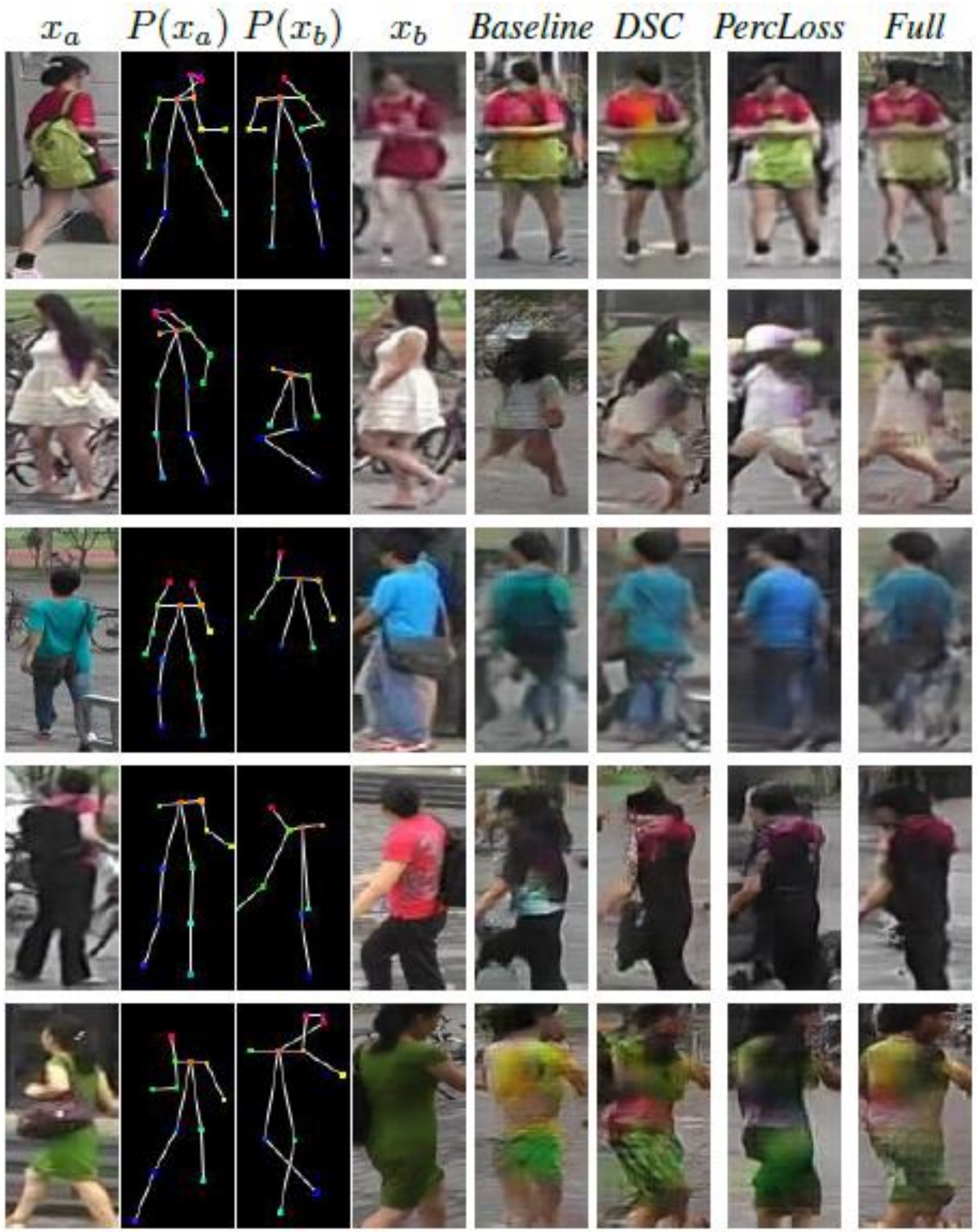
Conditional Image Generation



Qualitative results on the Market-1501 dataset



Qualitative results on the DeepFashion dataset



Badly generated images

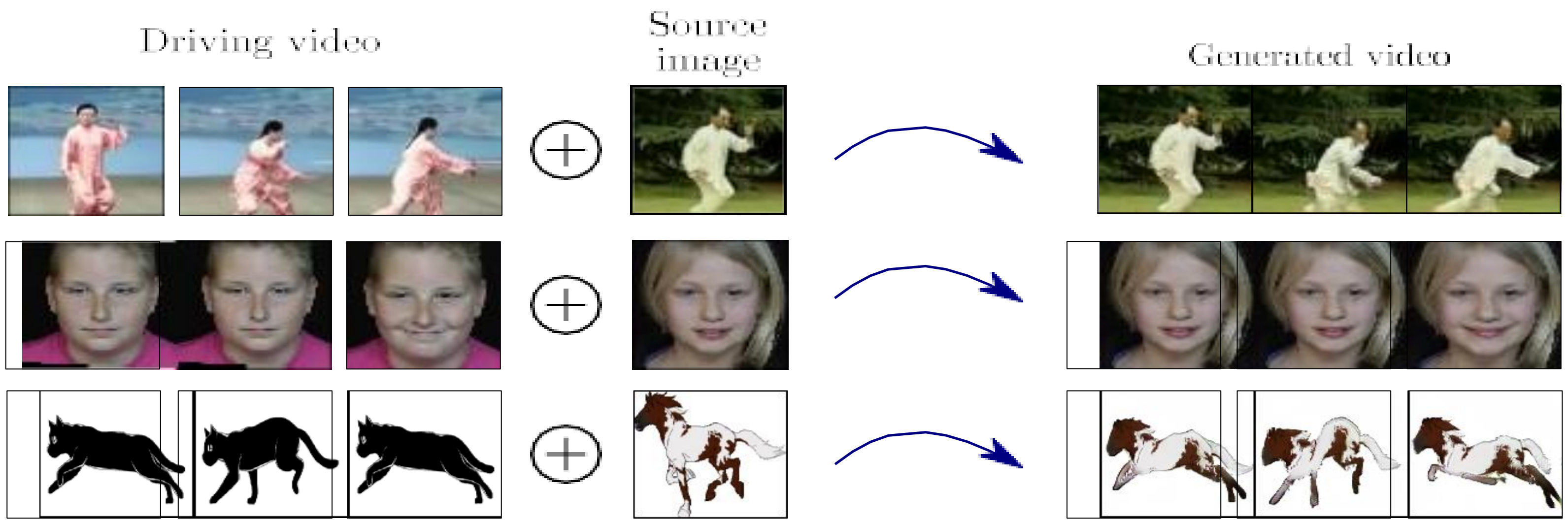
- errors of the pose estimation
- ambiguity of the pose estimation
- rare object appearance
- rare poses

Image Animation

- Siarohin, et al., “Animating Arbitrary Objects via Deep Motion Transfer”, CVPR19
- Siarohin, et al., “First Order Motion Model for Image Animation”, NeurIPS19

https://github.com/AliaksandrSiarohin/first-order-model

Image Animation: Appearance or Motion Transfer?



Appearance transfer

Detect pose in each frame of the driving video

Apply our pose-base image generator with the source image and each detected pose

Problems: requires a detector, does not work when the shapes of the object are different (ie. short to tall persons) => **Use Unsupervised Transfer Motion**

Image Animation with MOviNg KEYpoints

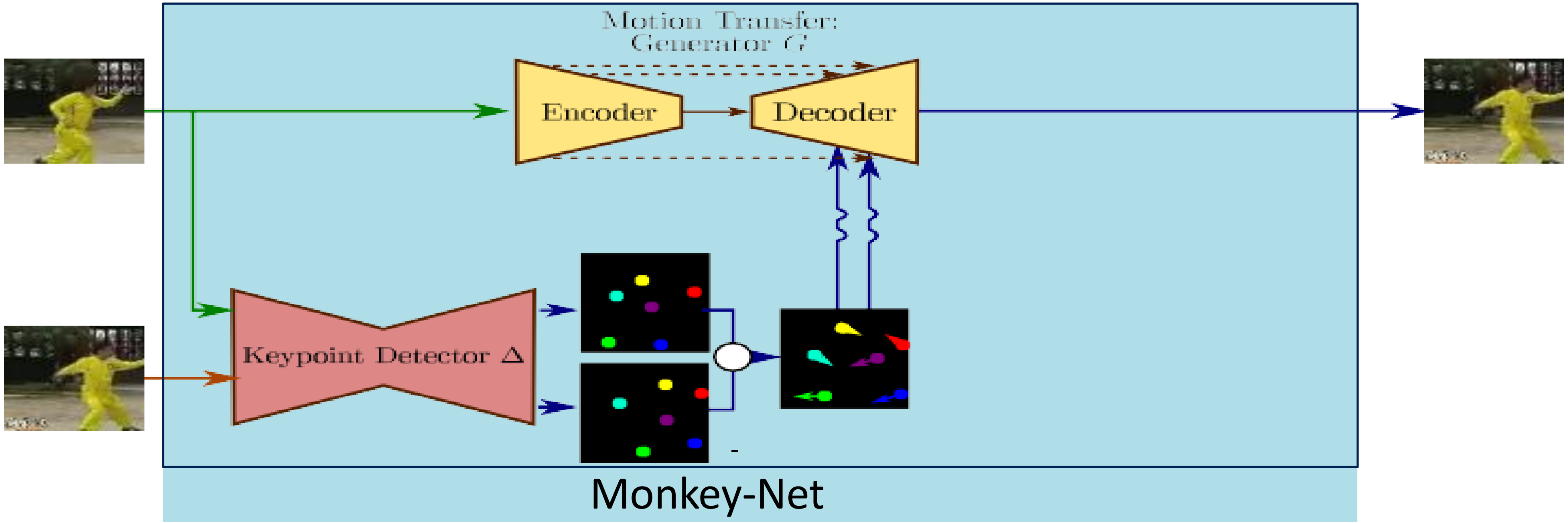
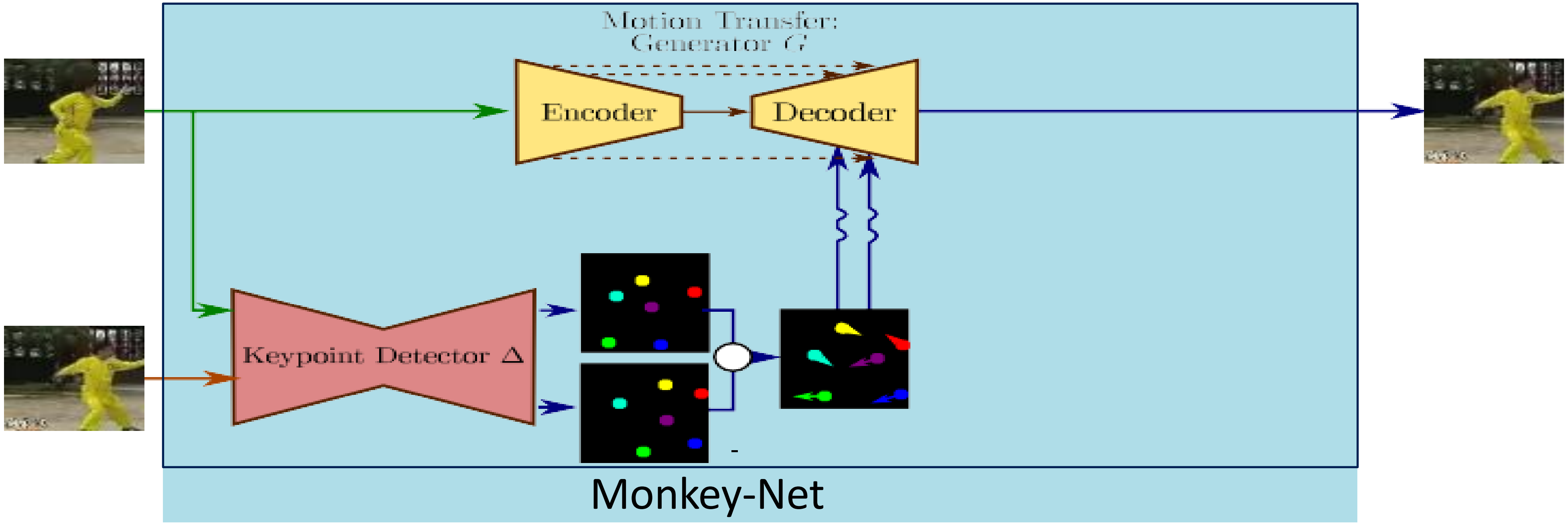
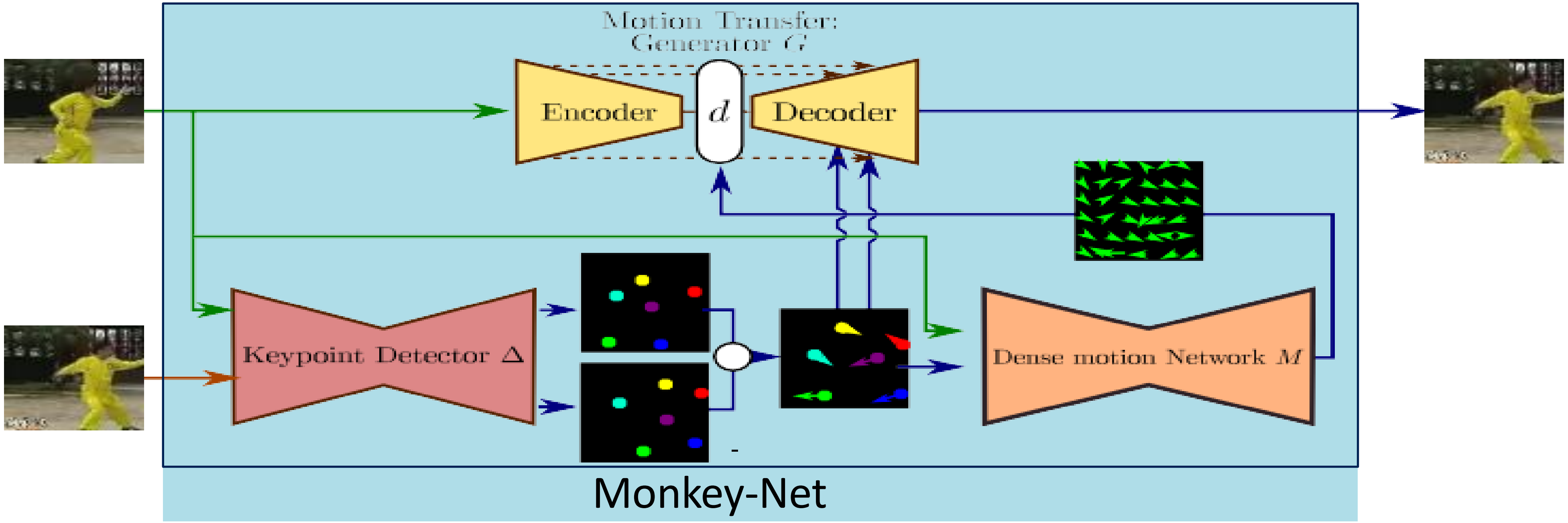


Image Animation with MOviNg KEYpoints



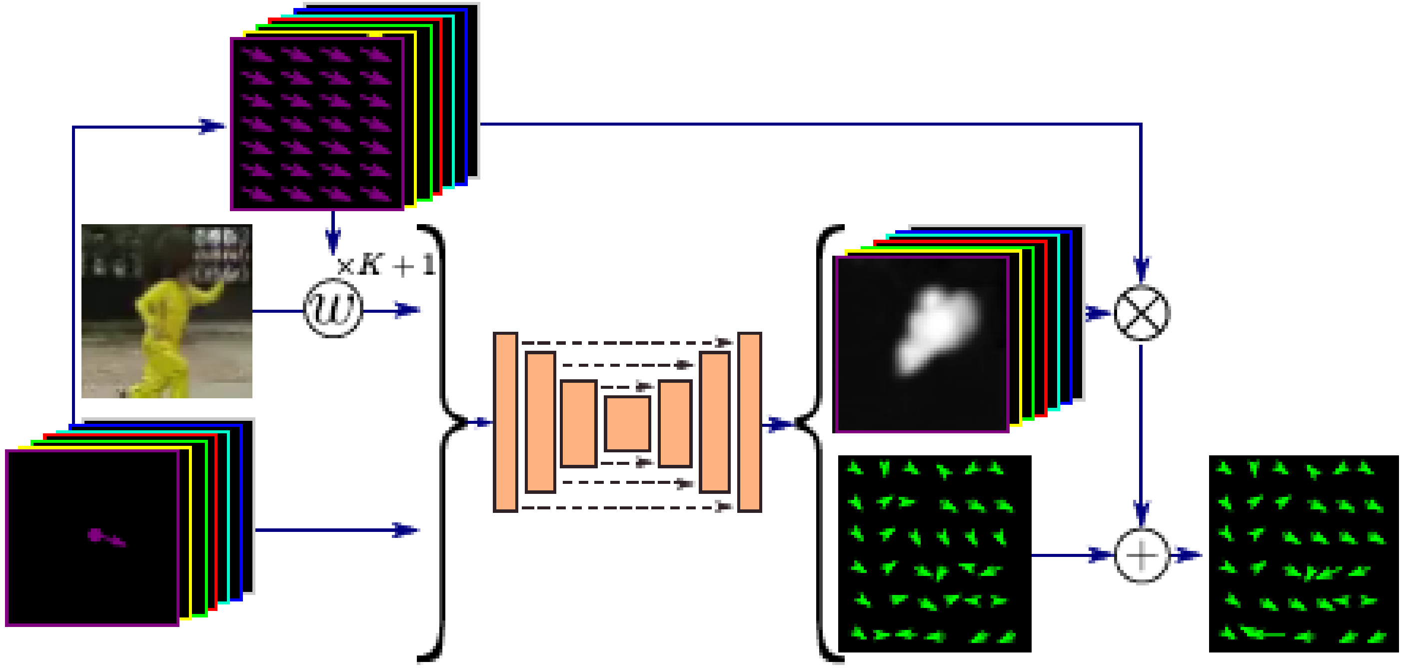
Again, we have an alignment problem

Image Animation with MOviNg KEYpoints



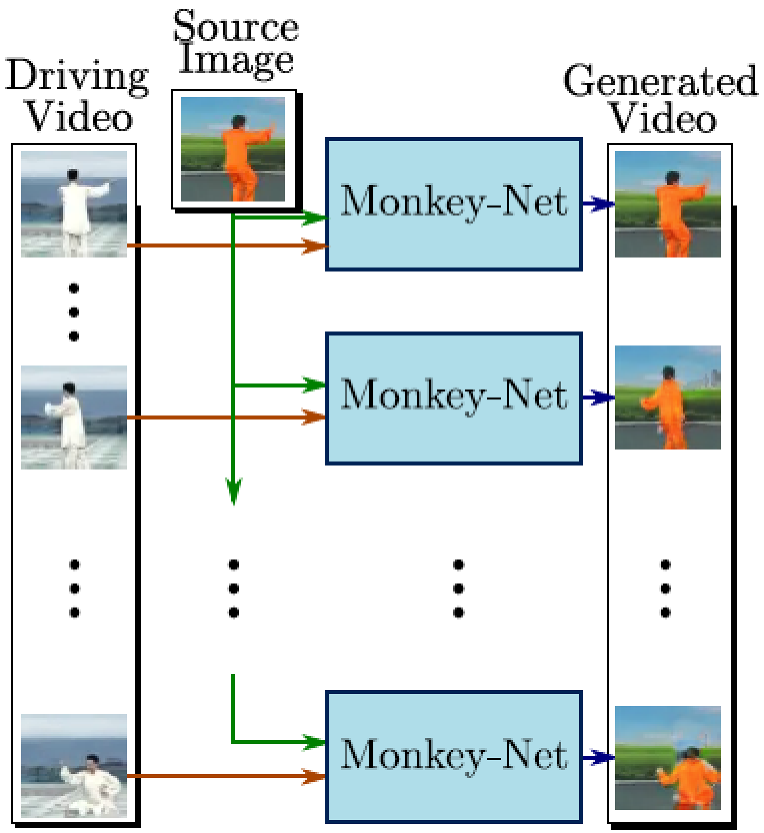
- Monkey-Net has a motion-specific keypoint detector Δ, a motion prediction network M, and an image generator G (reconstructs the image x′ from the keypoint positions Δ(x) and Δ(x′)); Optical flow computed by M is used by G to handle misalignments between x and x′
- The model is learned with a self-supervised learning scheme

Image Animation: Motion Prediction



From the appearance of the first frame and the keypoints motion, the network M predicts a mask for each keypoint and the residual motion

Image Animation Generation



At testing time the model generates a video with the object appearance of the source image but with motion from driving video:

- transfer the motion between the source image and each driving frame
- provide the generator the relative difference between keypoints

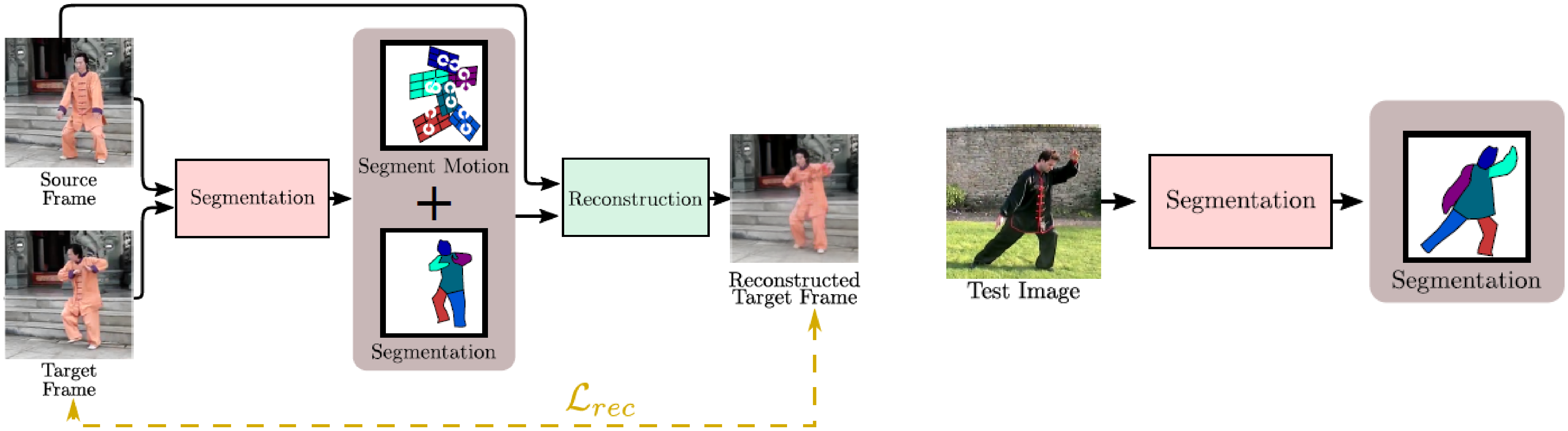
Learned Keypoints

Motion-supervised Co-Part Segmentation

- Siarohin, et al., “Motion Supervised Co-Part Segmentation”, ICPR20

https://github.com/AliaksandrSiarohin/motion-cosegmentation

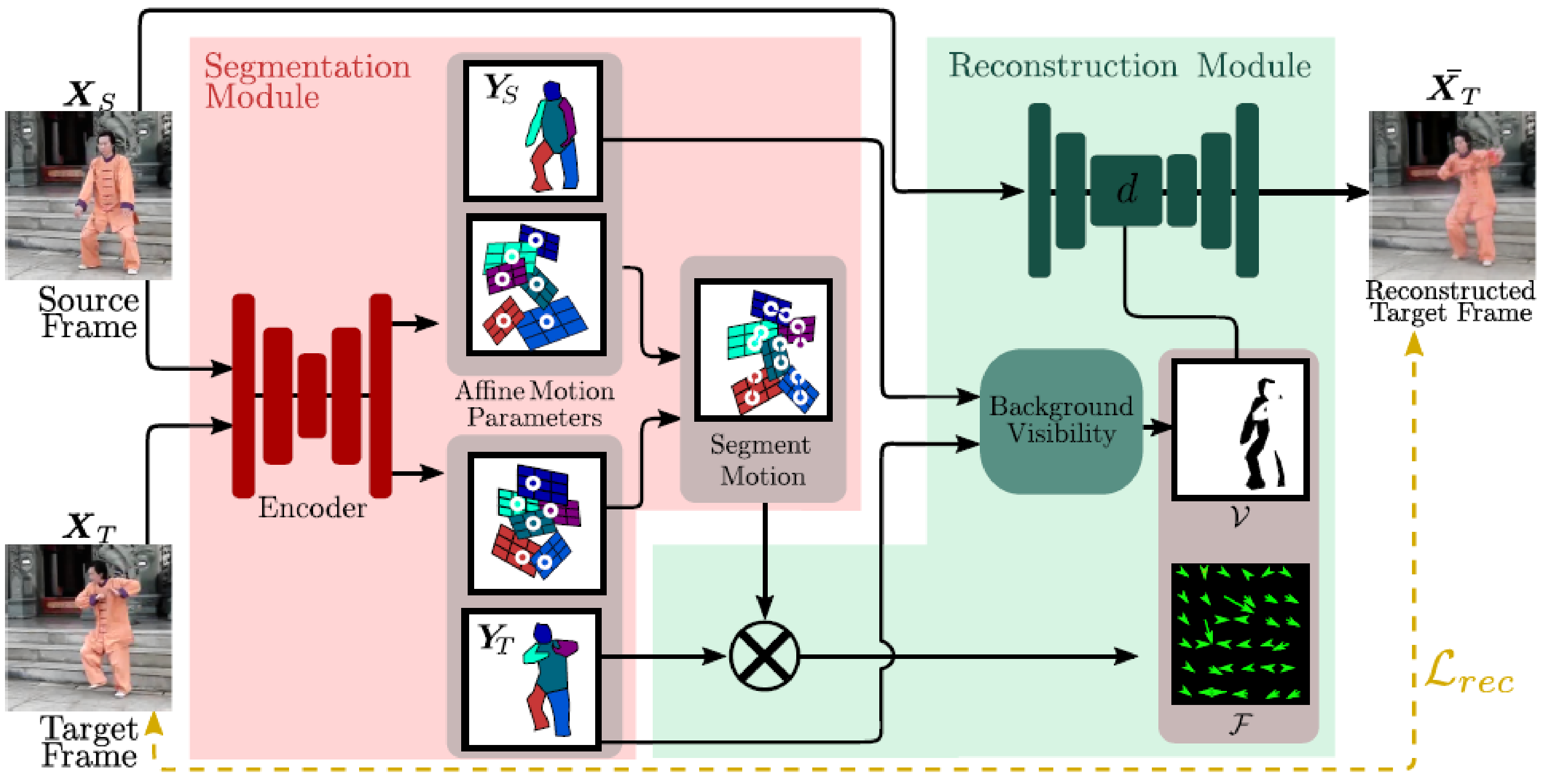
Self-supervised Co-Part Segmentation



Leverage motion info to train a segmentation network without annotation

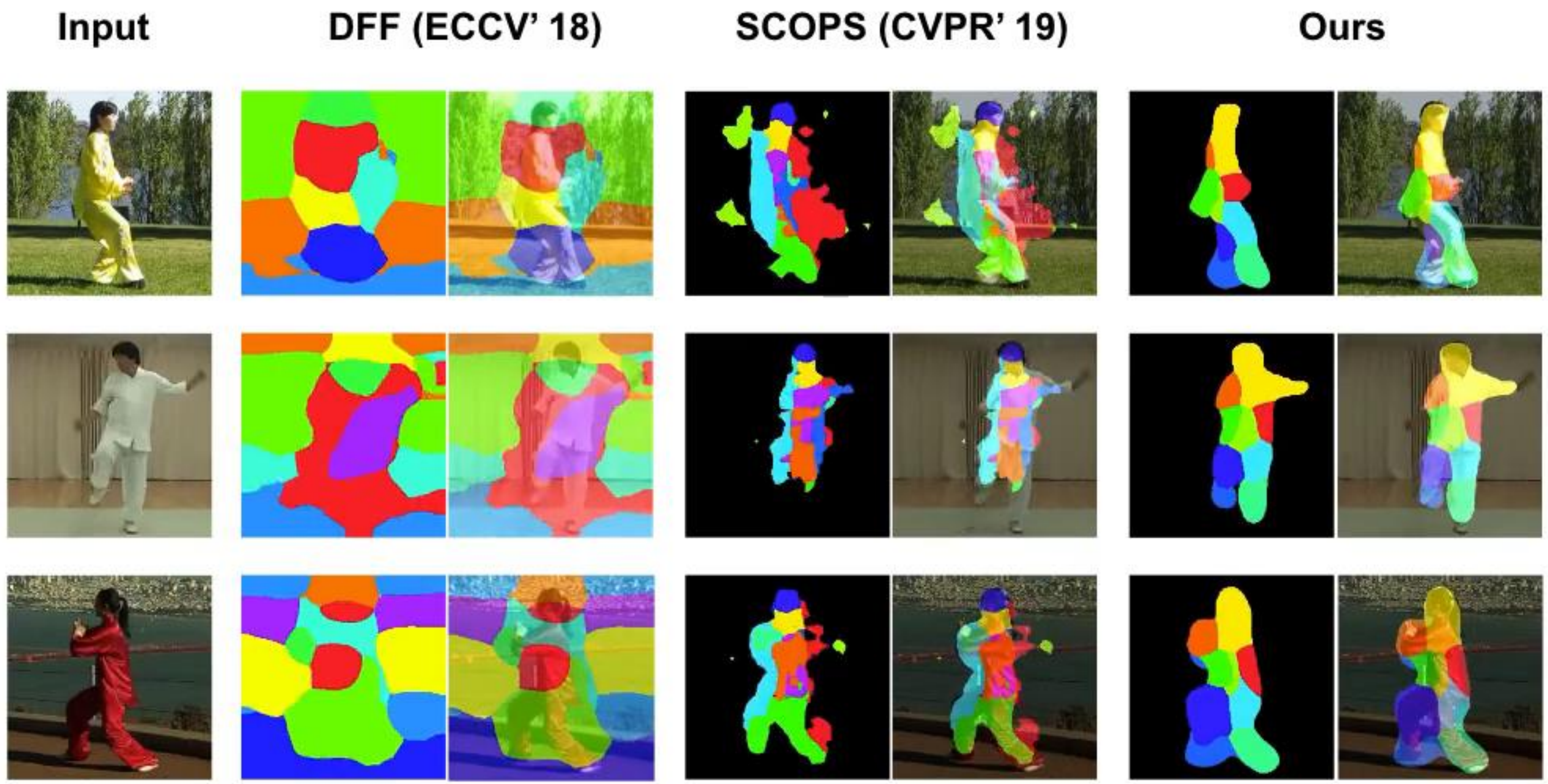
- At training, use frame pairs (source and target) extracted from the same video => predict segments in target that can be combined with a motion representation between the two frames to reconstruct the target frame
- At inference, use the trained segmentation model to predict object parts segments

Self-supervised Co-Part Segmentation



- **Segmentation Module** predicts the segmentation maps Y_S and Y_T, and the affine motion parameters
- **Reconstruction Module:** (1) computes a background visibility mask V and an optical flow F; (2) reconstructs the target frame X_T by warping the features of the source frame X_S and masking occluded features

Tai-Chi-HD

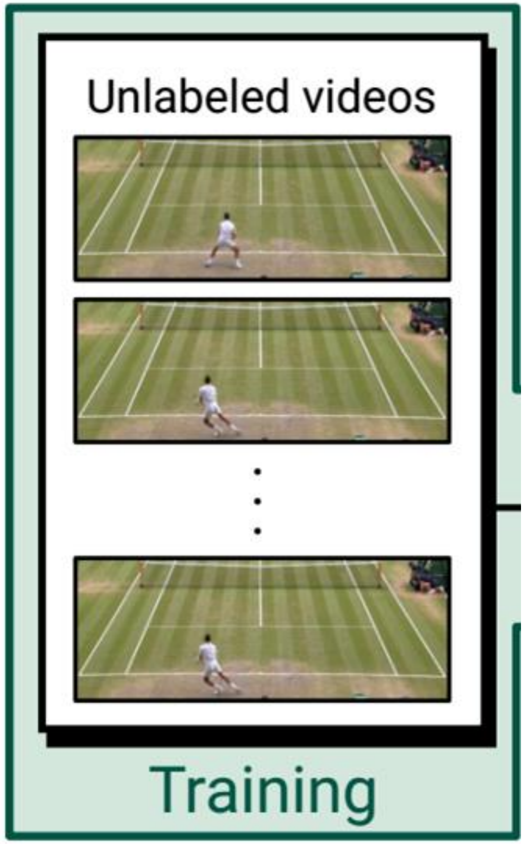


Playable Video Generation

- Menapace, et al., “Playable Video Generation”, CVPR21

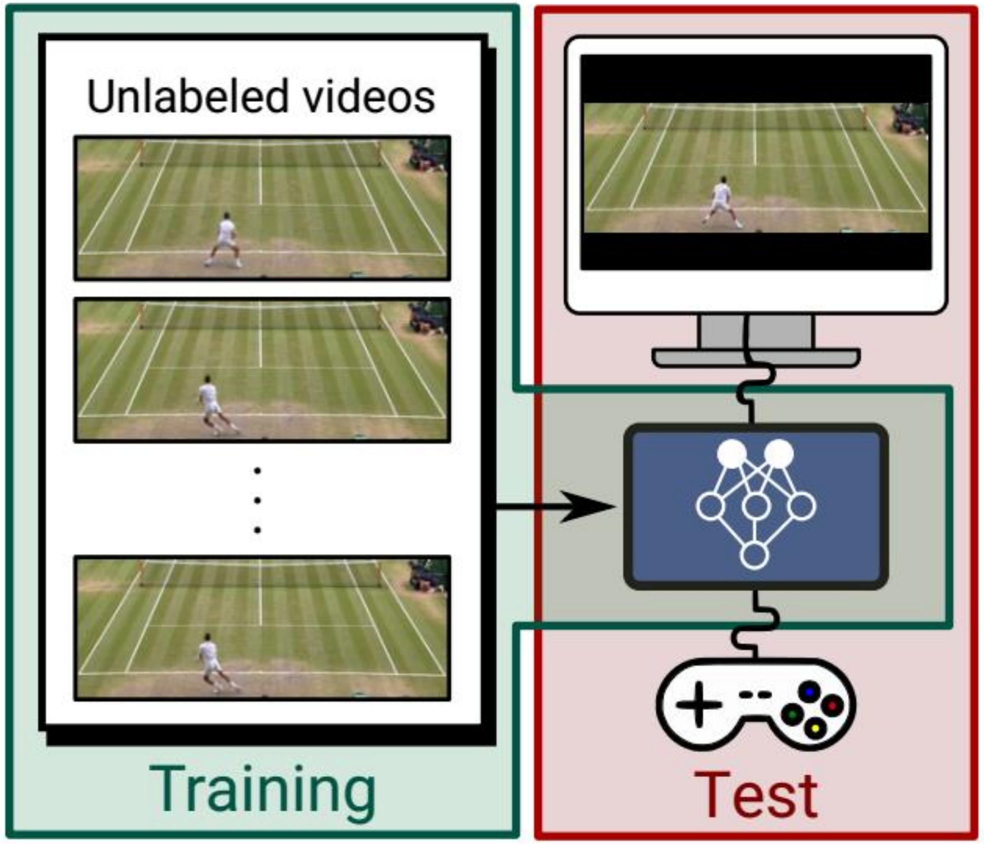
https://github.com/willi-menapace/PlayableVideoGeneration

Playable Video Generation



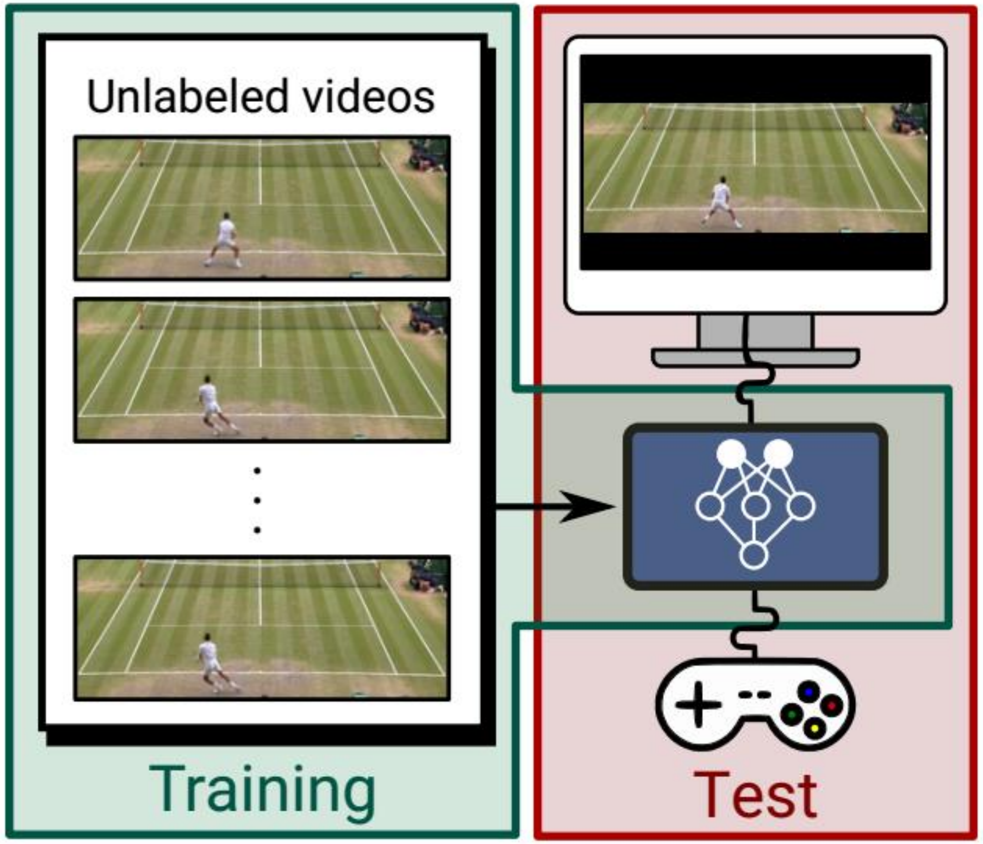
- Consider a set of videos depicting an agent acting in an environment
- Differently from other methods that use frame by frame action annotations, we assume no annotation is present

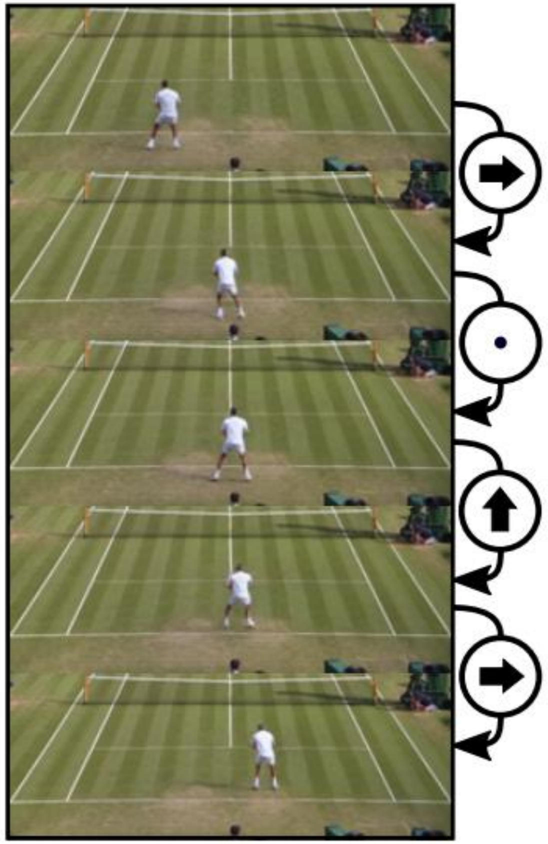
Playable Video Generation



- Learn a model that represents the observed environment.
- Allow the user to input actions to the model through a controller at test time

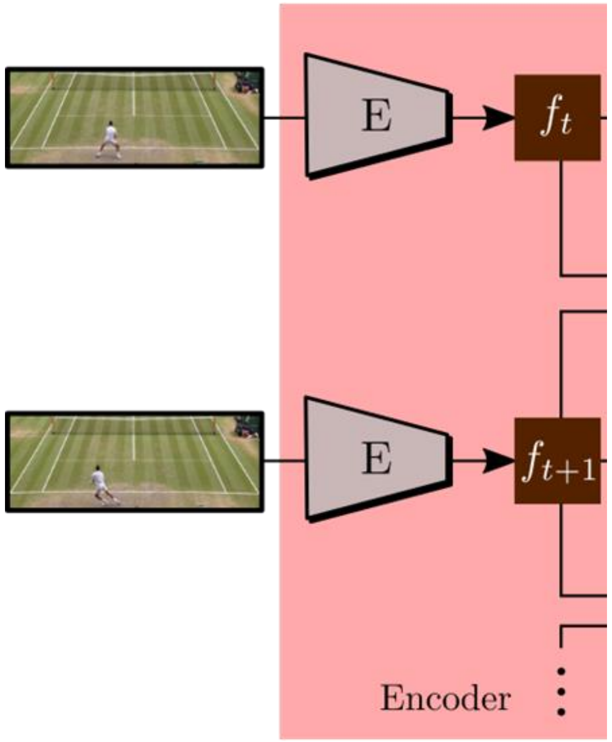
Playable Video Generation





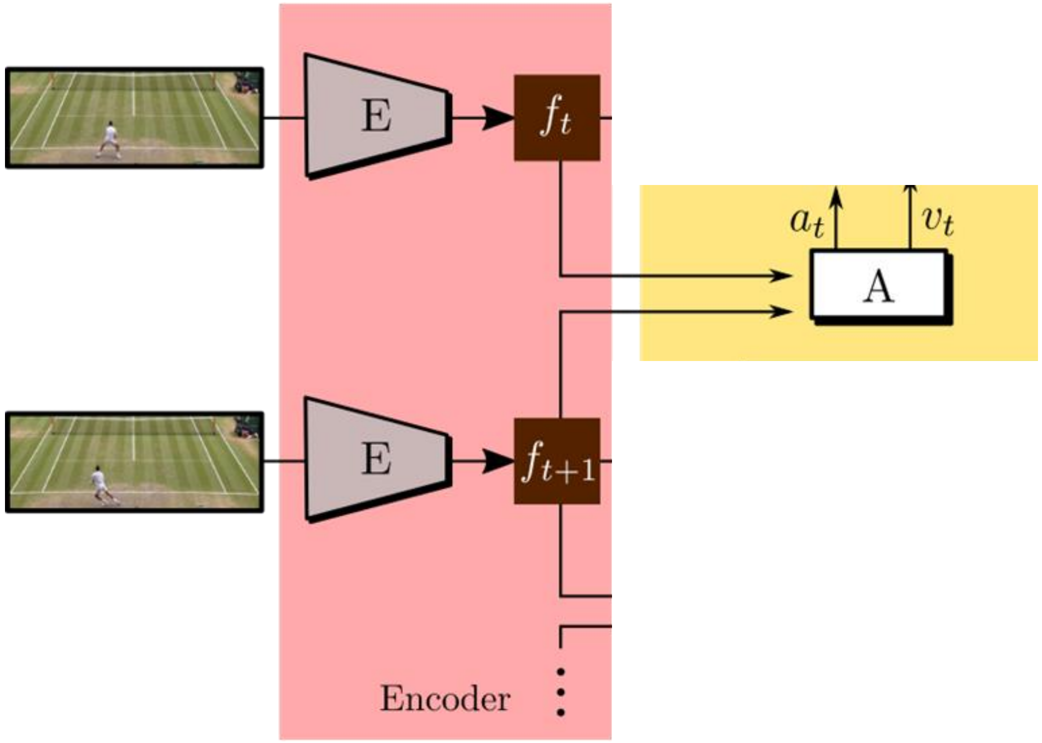
- Produce a video where the agent acts according to the actions specified by the user

Architecture



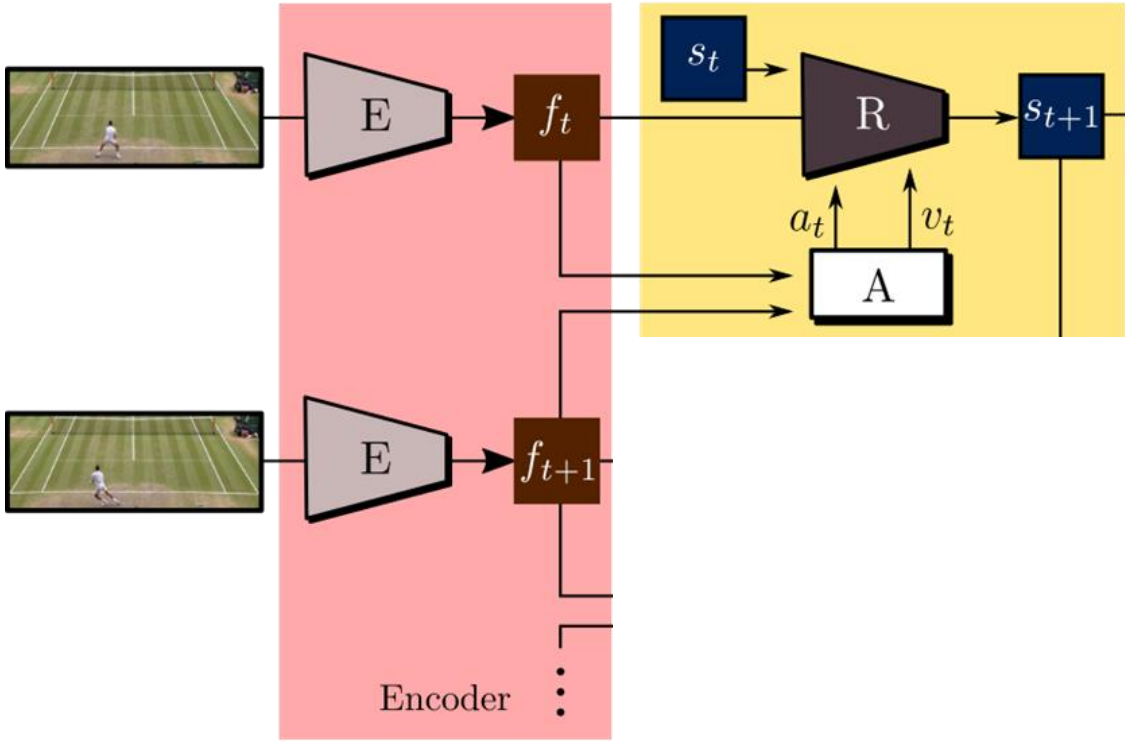
- First we sample an input sequence and use an encoder network to extract frame features

Architecture



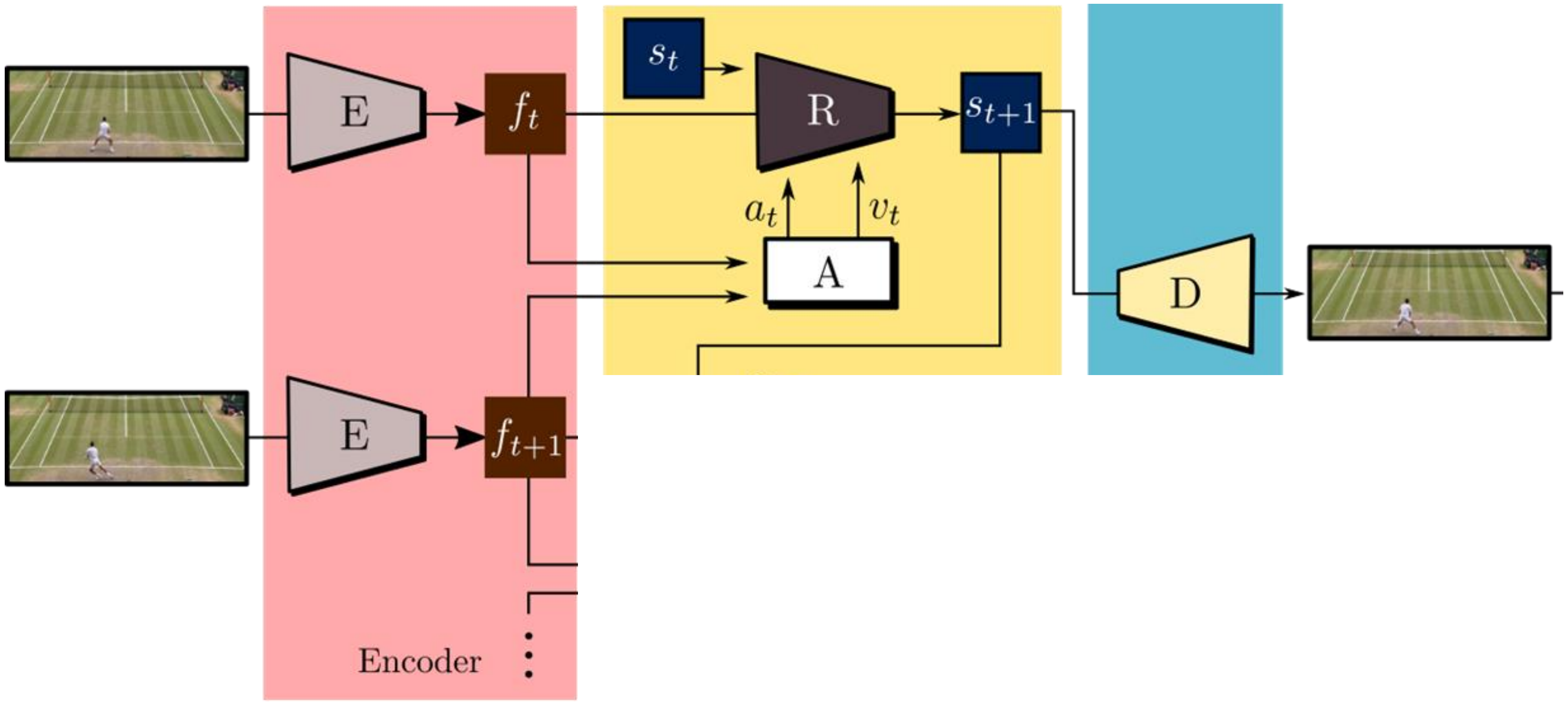
- Use then pairs of successive features to infer the action that was performed by the agent in the corresponding transition using an action network

Architecture



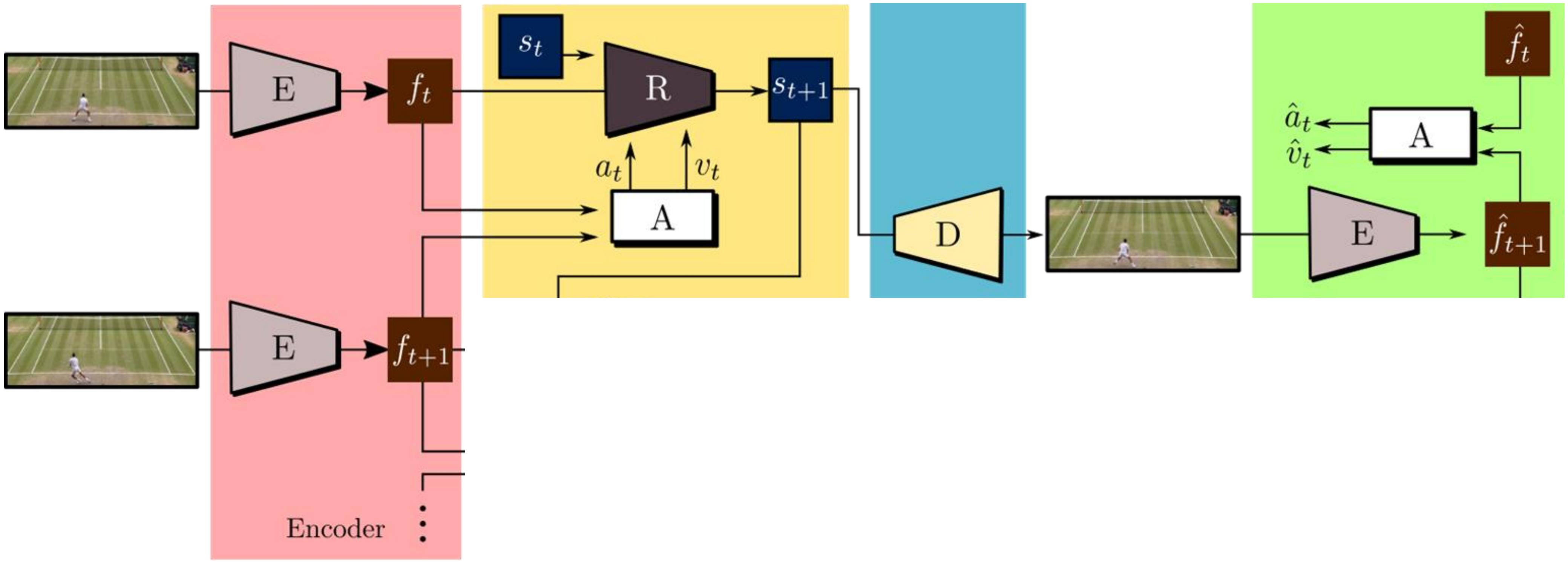
- Given the frame features and the action, a recurrent model is used to produce features representing the successive state

Architecture



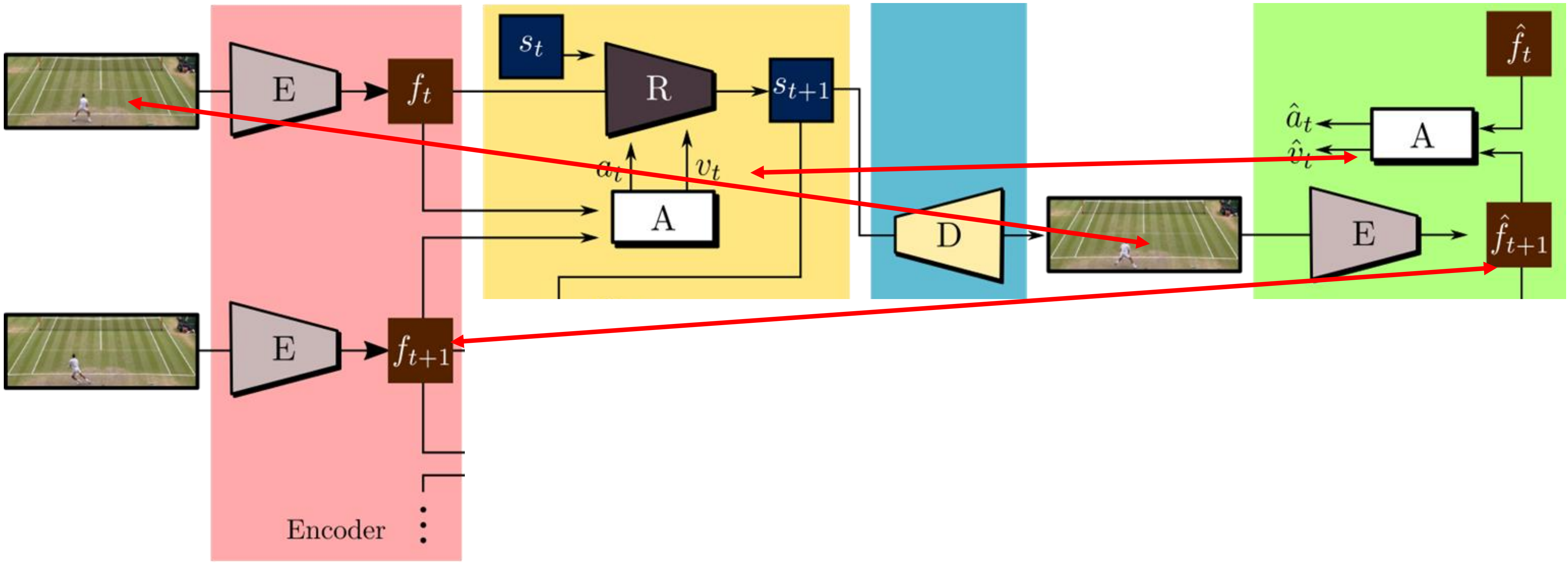
- The successive state is translated back to an image using a decoder network

Architecture



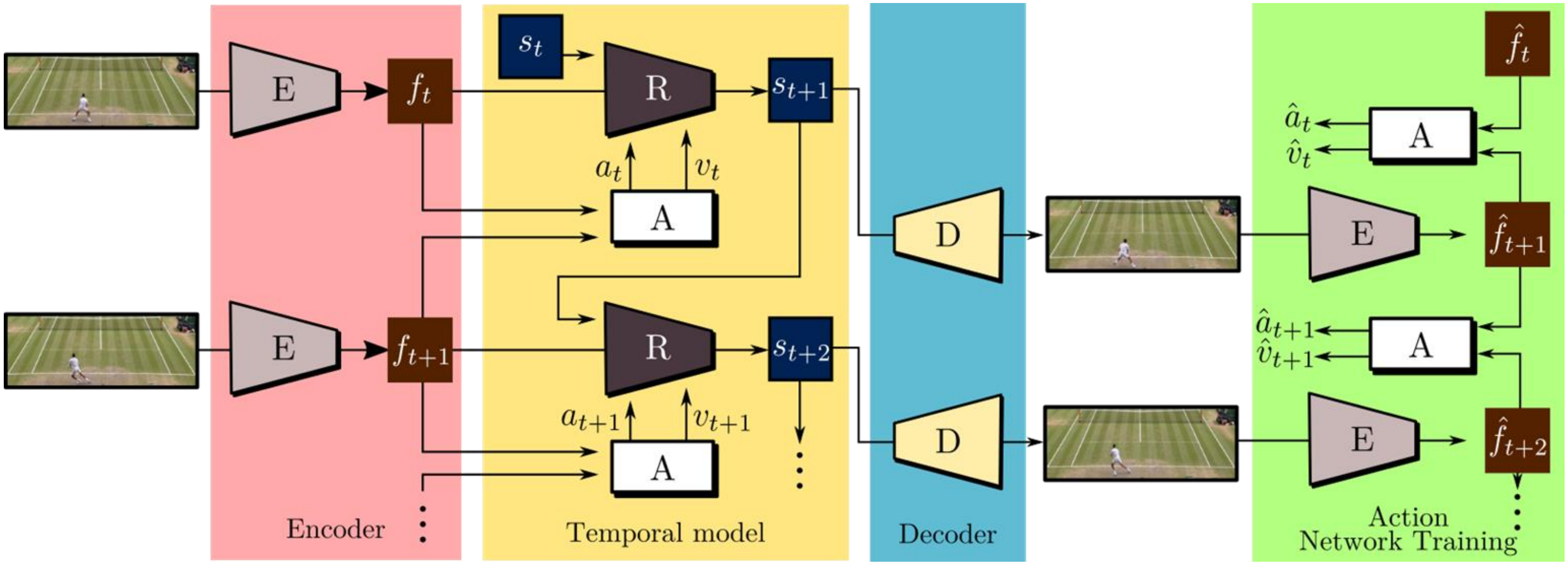
- For extra supervision, we encode back the produced frame using the encoder and the action network

Architecture



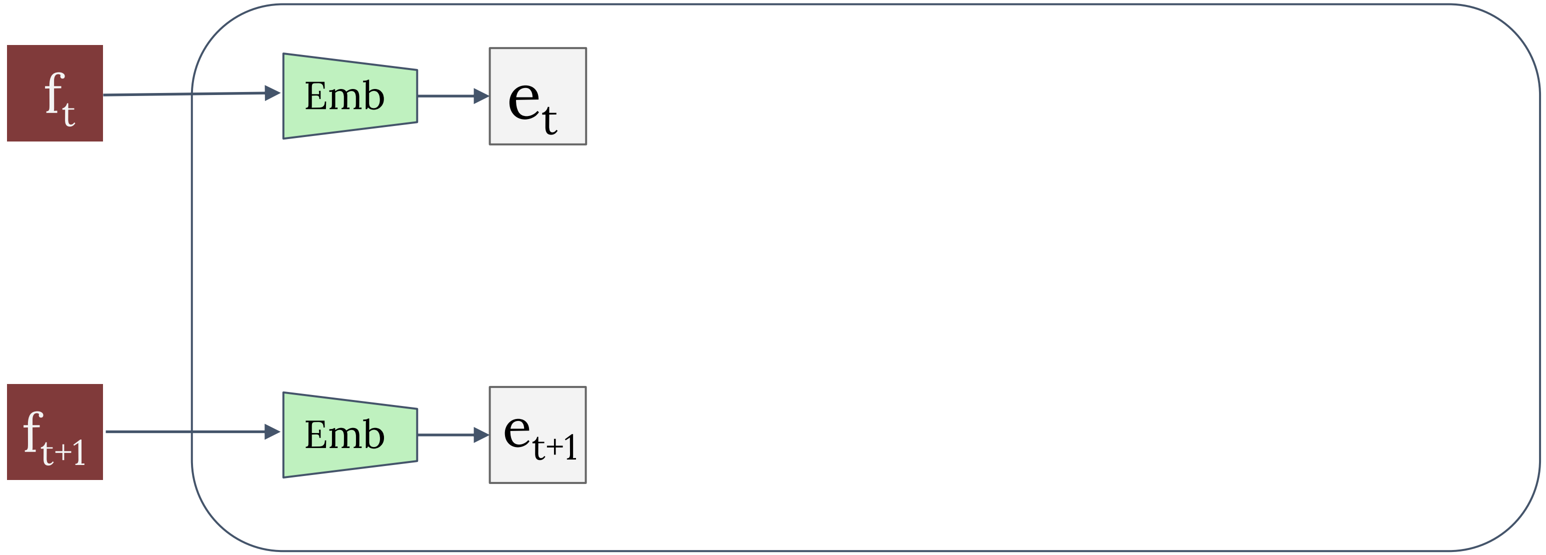
- Impose different self supervision losses on the frames, the frame features and the produced actions: use a mutual information maximization loss between actions and reconstructed actions as the main driving loss for action learning

Architecture



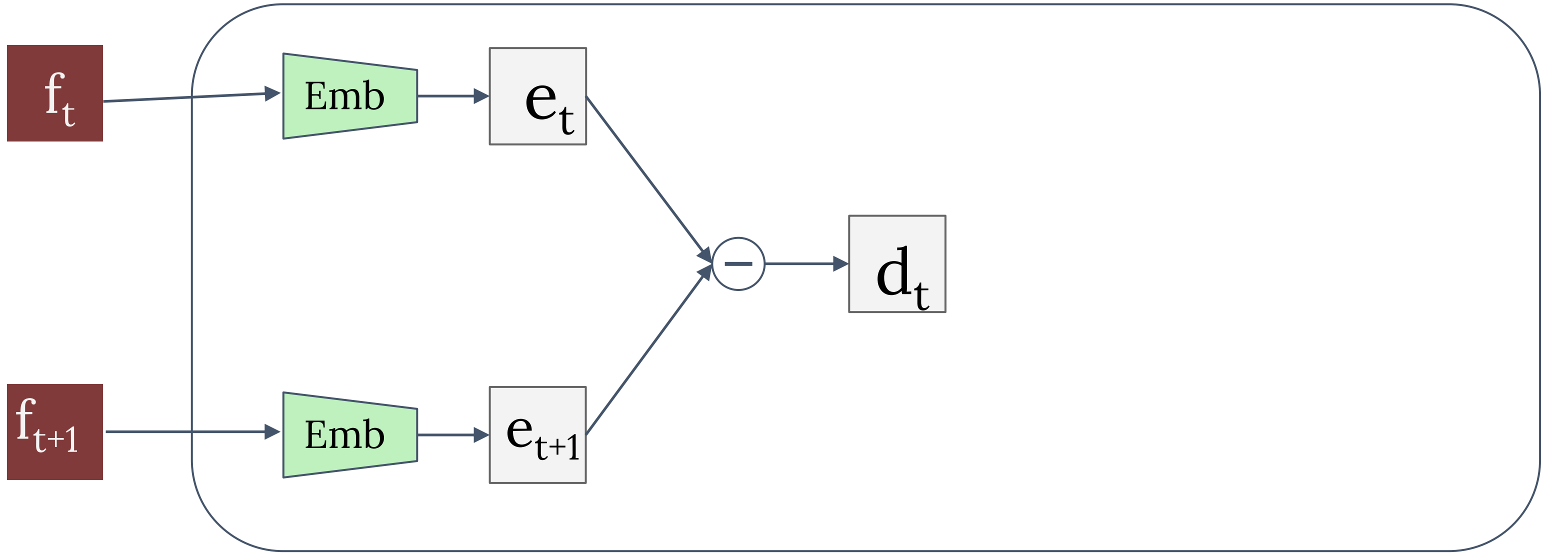
- The model is then unrolled over the whole sequence

Action Network



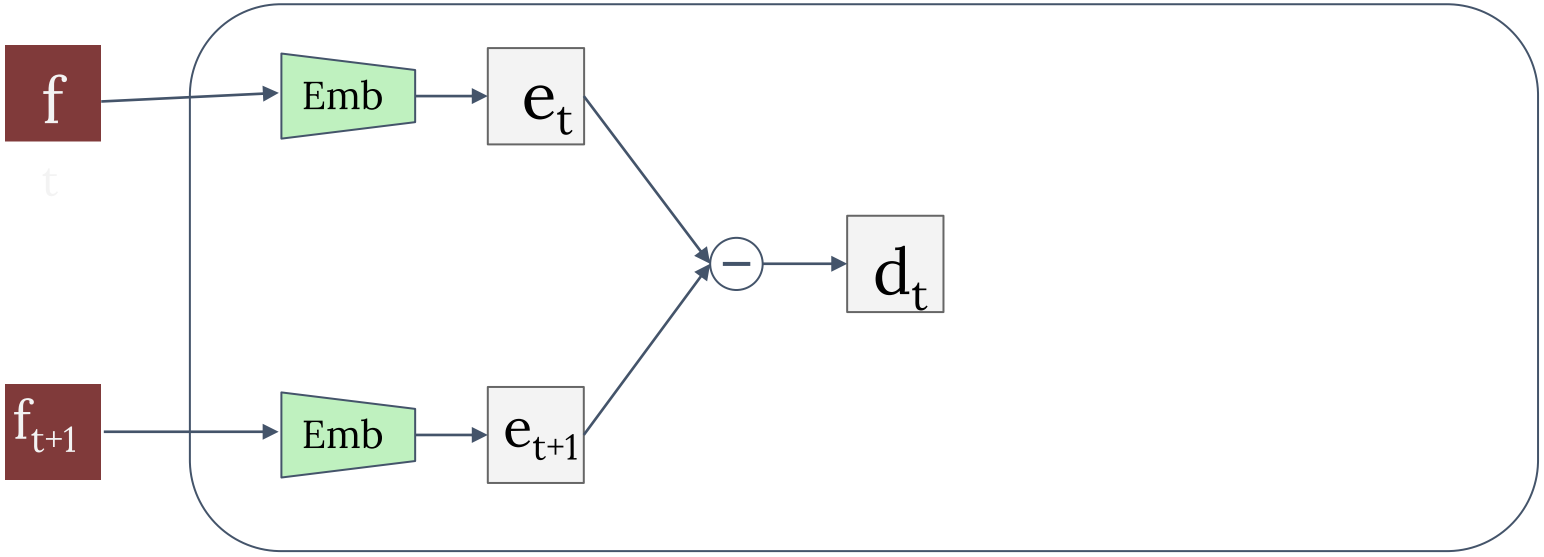
- The action network first encodes the frame features using a Multi Layer Perceptron to produce two embeddings

Action Network

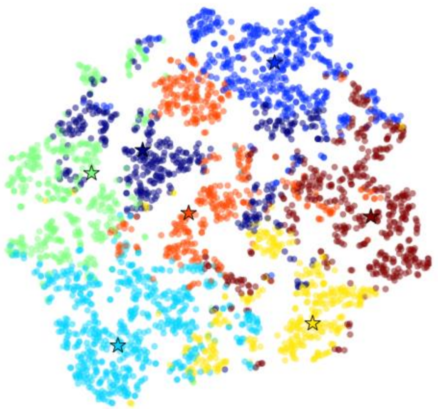


- We take the difference between these embedding as the representation of the transition between two frames: action direction d_t

Action Network



t-SNE plot of d_t

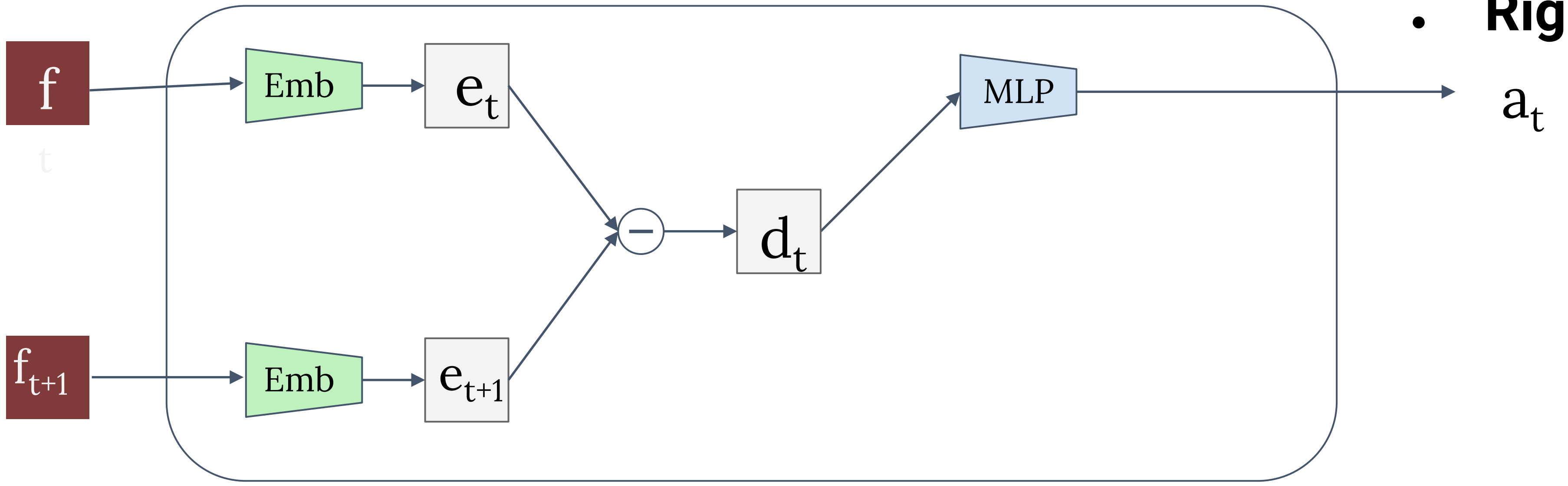


- When visualized, the learned space of action directions is a representation of the different types of transitions that are observed in the training videos

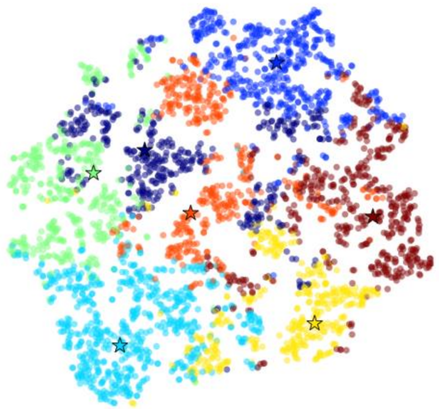
Action Network

Which action is done

- **Left**
- **Right**



t-SNE plot of d_t



- Use an MLP to assign a label to each point d_t: the high-level action associated to the current frame
- Use of action variability embeddings to ensure a well-posed reconstruction loss on the frames

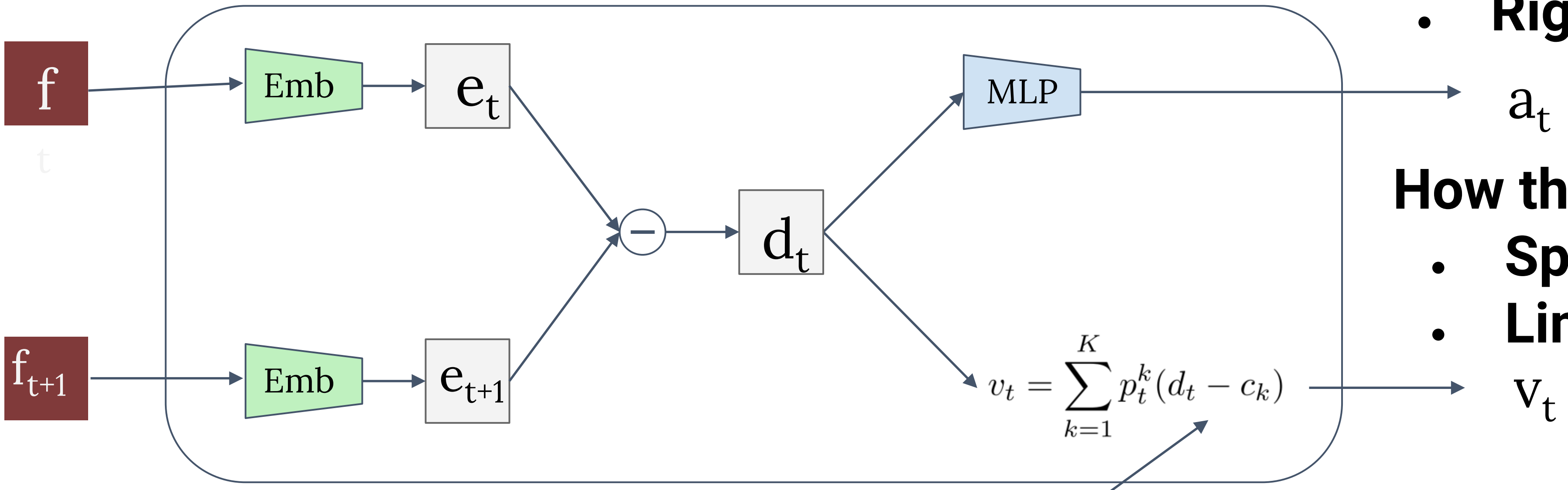
Action Network

Which action is done

- **Left**
- **Right**

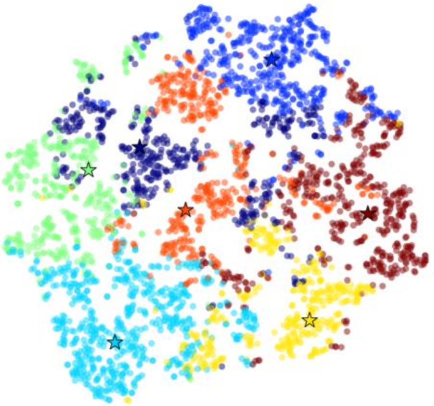
How the action is done

- **Speed**
- **Limb movement**



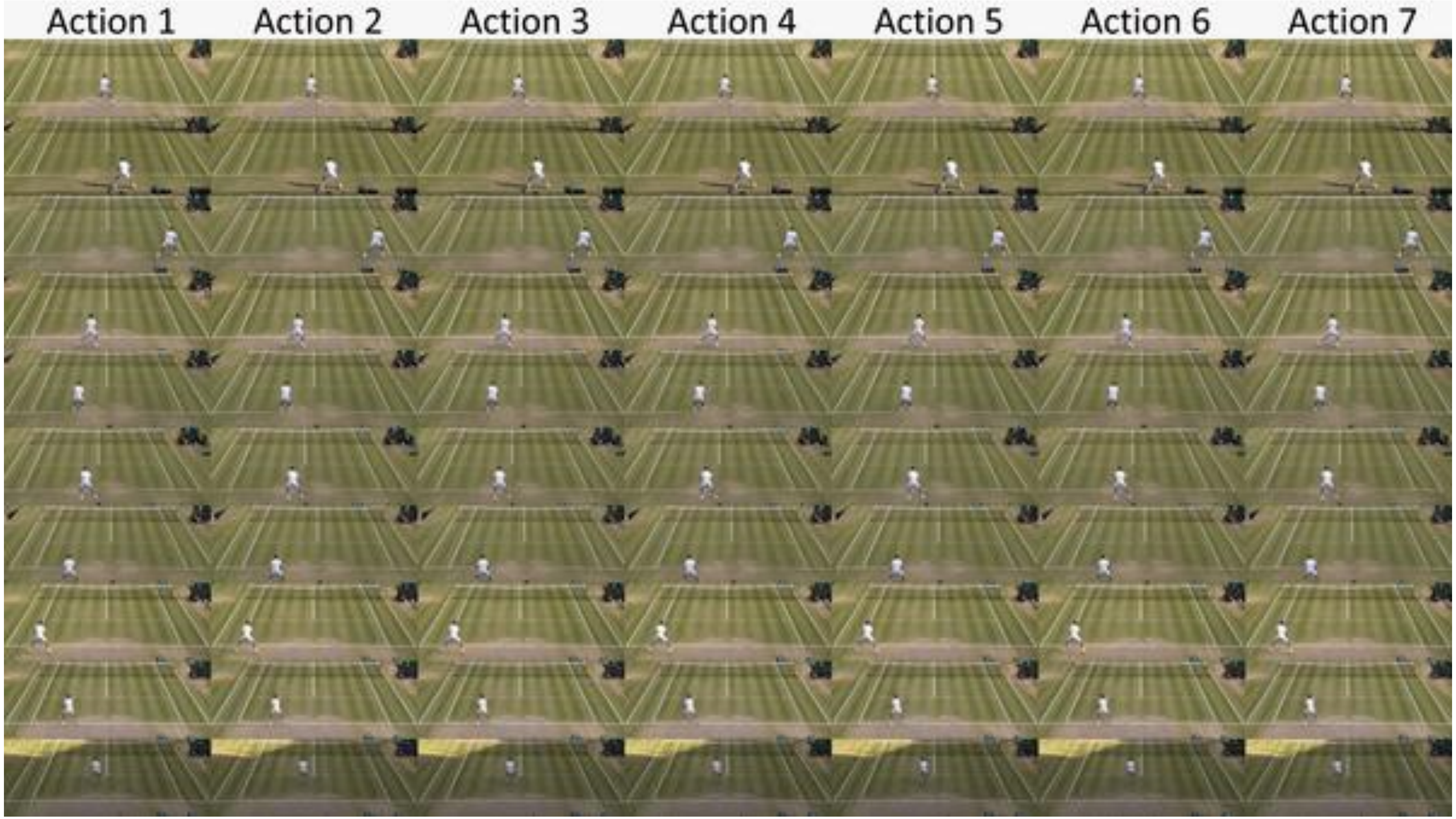
Expectation of distance from cluster centroids

t-SNE plot of d_t



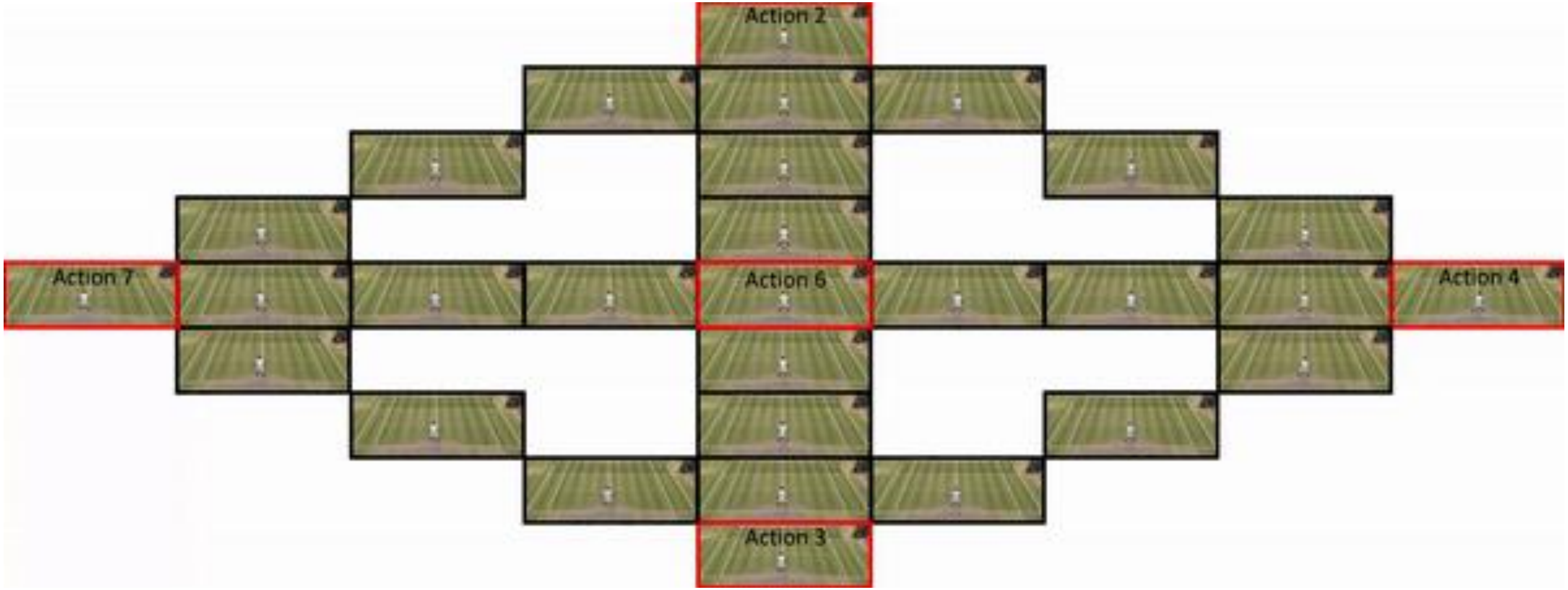
- For each d_t compute the expectation of its distance from the cluster centroids: variability embedding v_t => the specific way in which an action is performed

Results



- We learn a wide range of actions. The meaning of actions is consistent, independently from the starting frame the action is applied to

Action Interpolation



- At inference, we typically pose $v_t = 0$ and let the user specify actions a_t at each time step
- v_t can also be obtained from an action direction d_t that moves between the centroids of different actions: it is possible to generate a variety of different movement directions, eg. diagonal movements

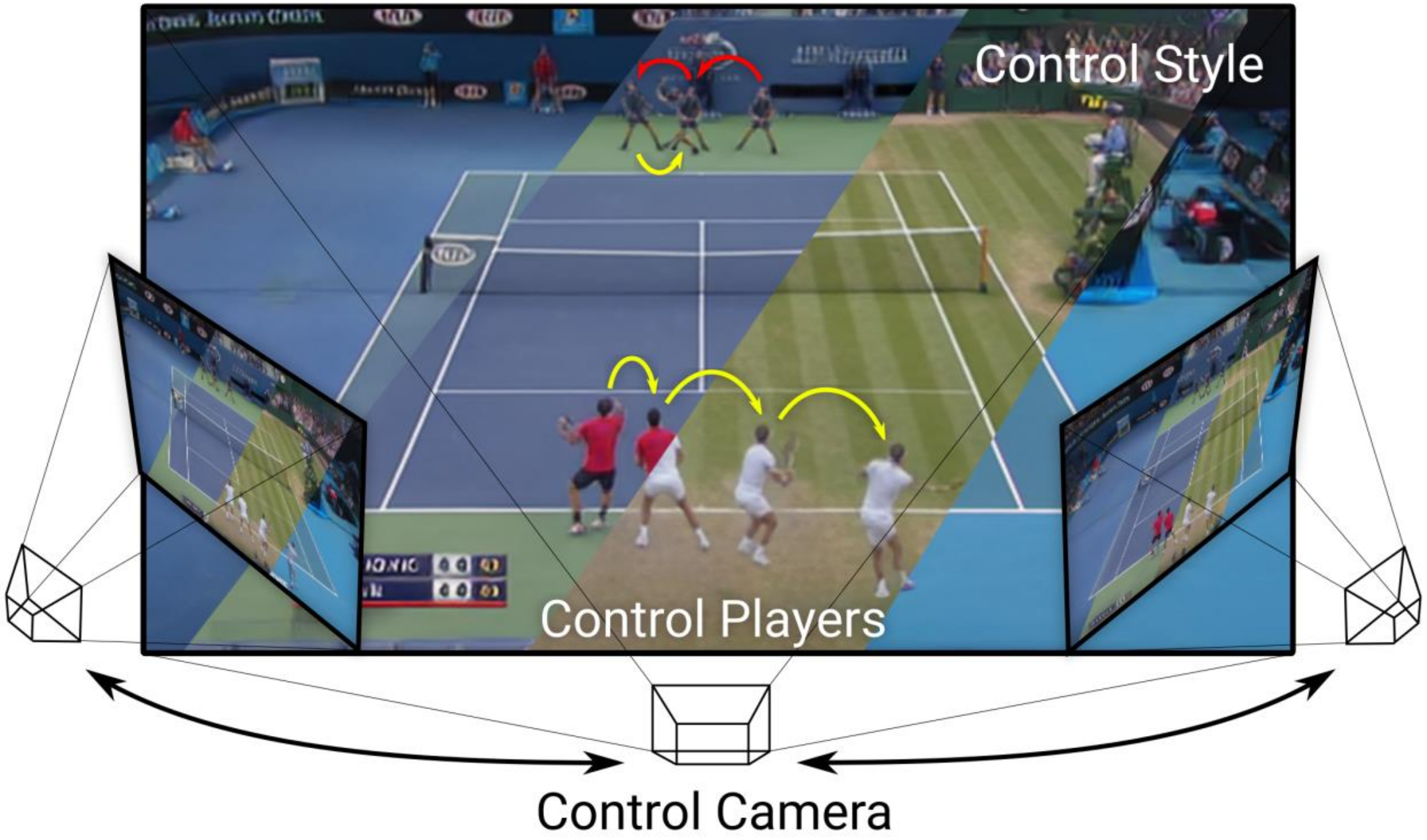

Action

Playable Environments

- Menapace, et al., “Playable Environments: Video Manipulation in Space and Time”, CVPR22

https://github.com/willi-menapace/PlayableEnvironments

Playable Environments



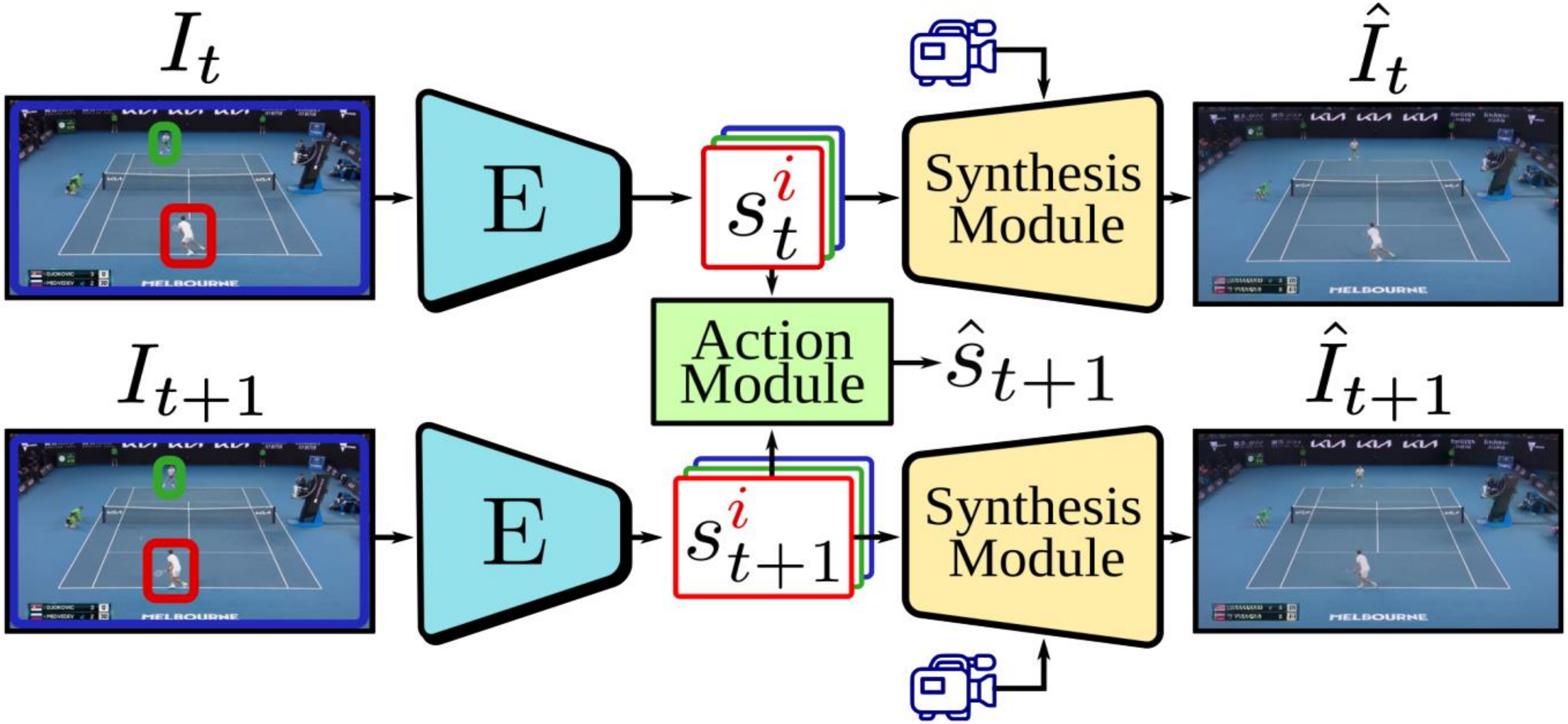
- Learn a model that represents the observed environment
- Allow the user to input actions to the model through a controller at test time

Playable Environments

Playable Environments

Playable Environments

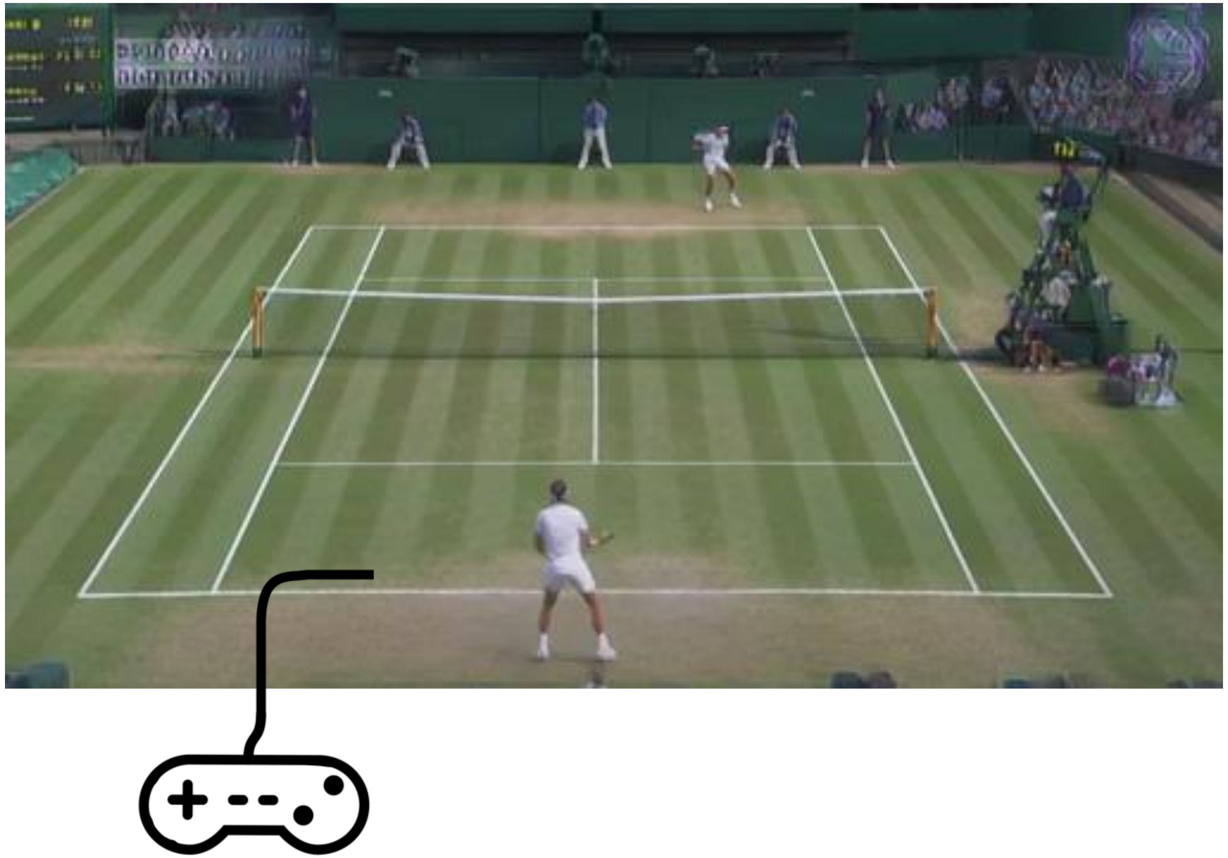
Framework



I_t
E
s_t^i
Synthesis Module
$\hat{I}_t$
Action Module
$\hat{s}_{t+1}$
I_{t+1}
E
s_{t+1}^i
Synthesis Module
$\hat{I}_{t+1}$

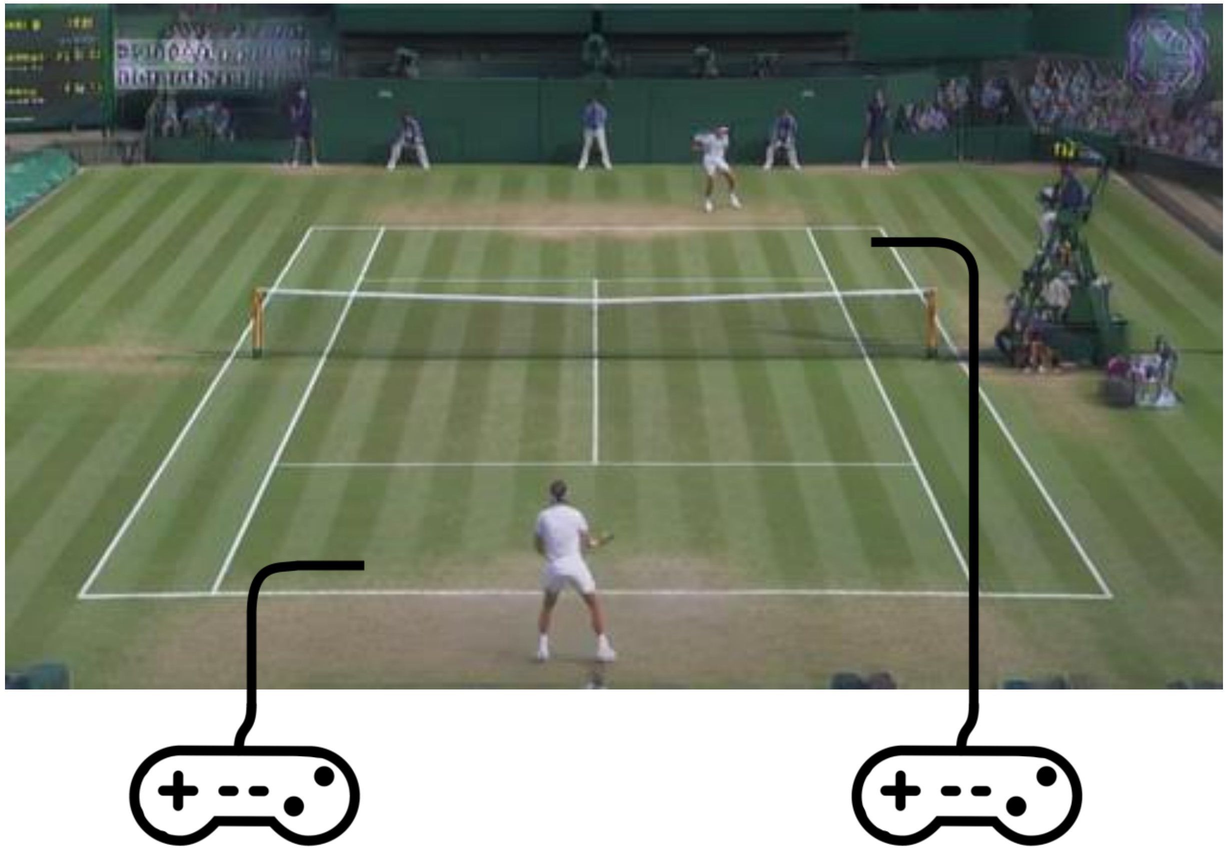
Framework Characteristics

1. Playability



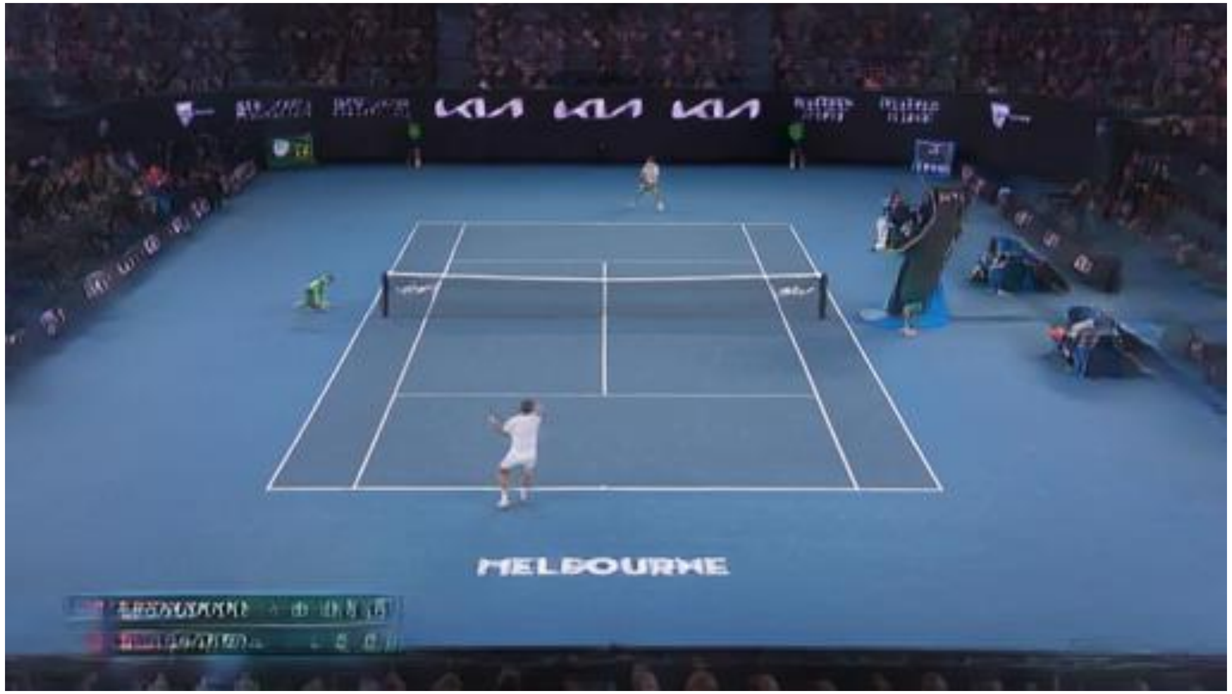
Framework Characteristics

1. Playability
2. Multi Object
3. Deformable Objects



Framework Characteristics

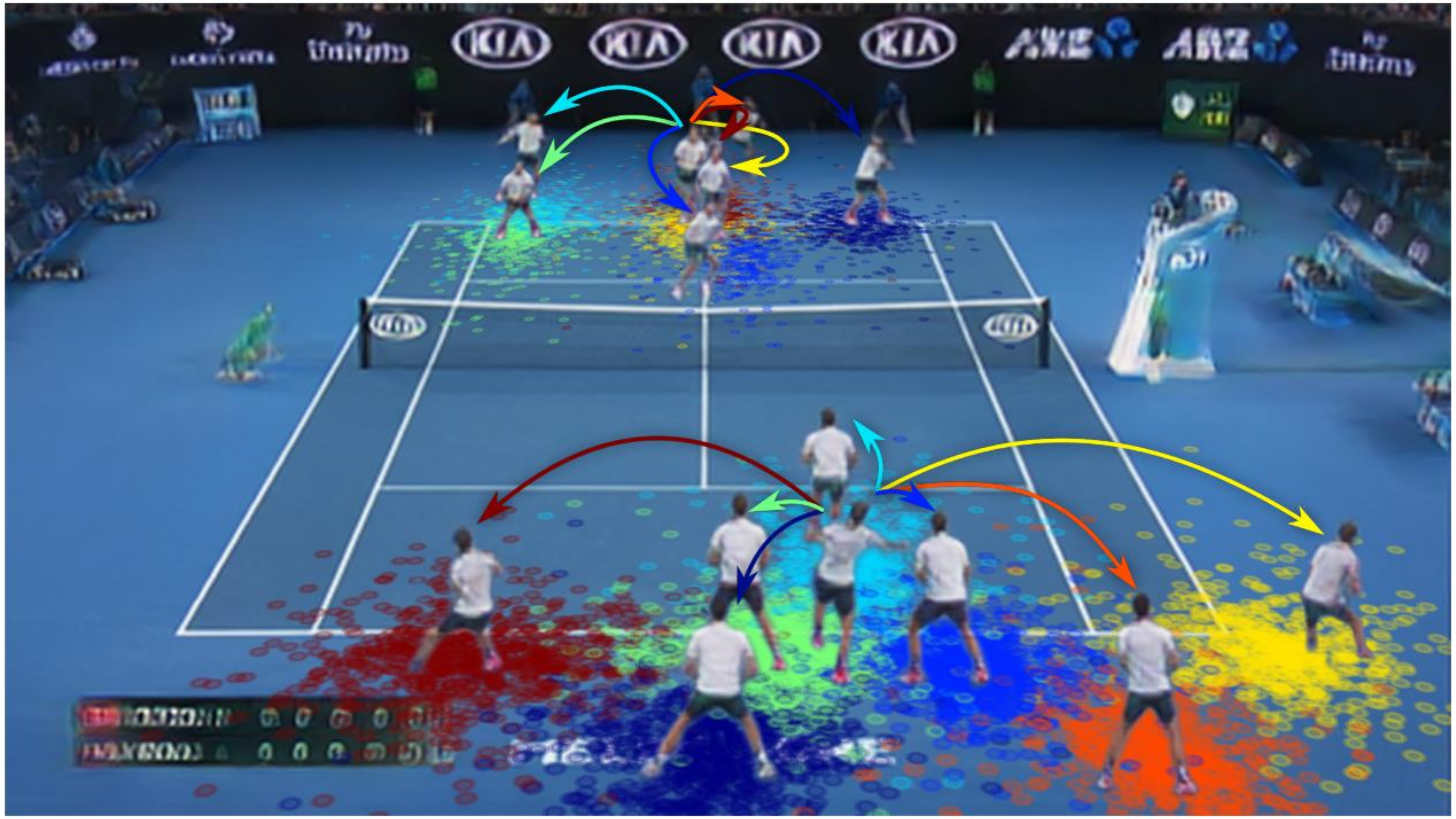
1. Playability
2. Multi Object
3. Deformable Objects
4. Camera Control



Framework Characteristics

1. Playability
2. Multi Object
3. Deformable Objects
4. Camera Control
5. Style Control
6. Robustness

Learned Actions



Learnable Game Engines (LGEs)

- Menapace, et al., “Plotting Behind the Scenes: Towards Learnable Game Engines”, arxiv 2023

https://learnable-game-engines.github.io/lge-website/

Related Work

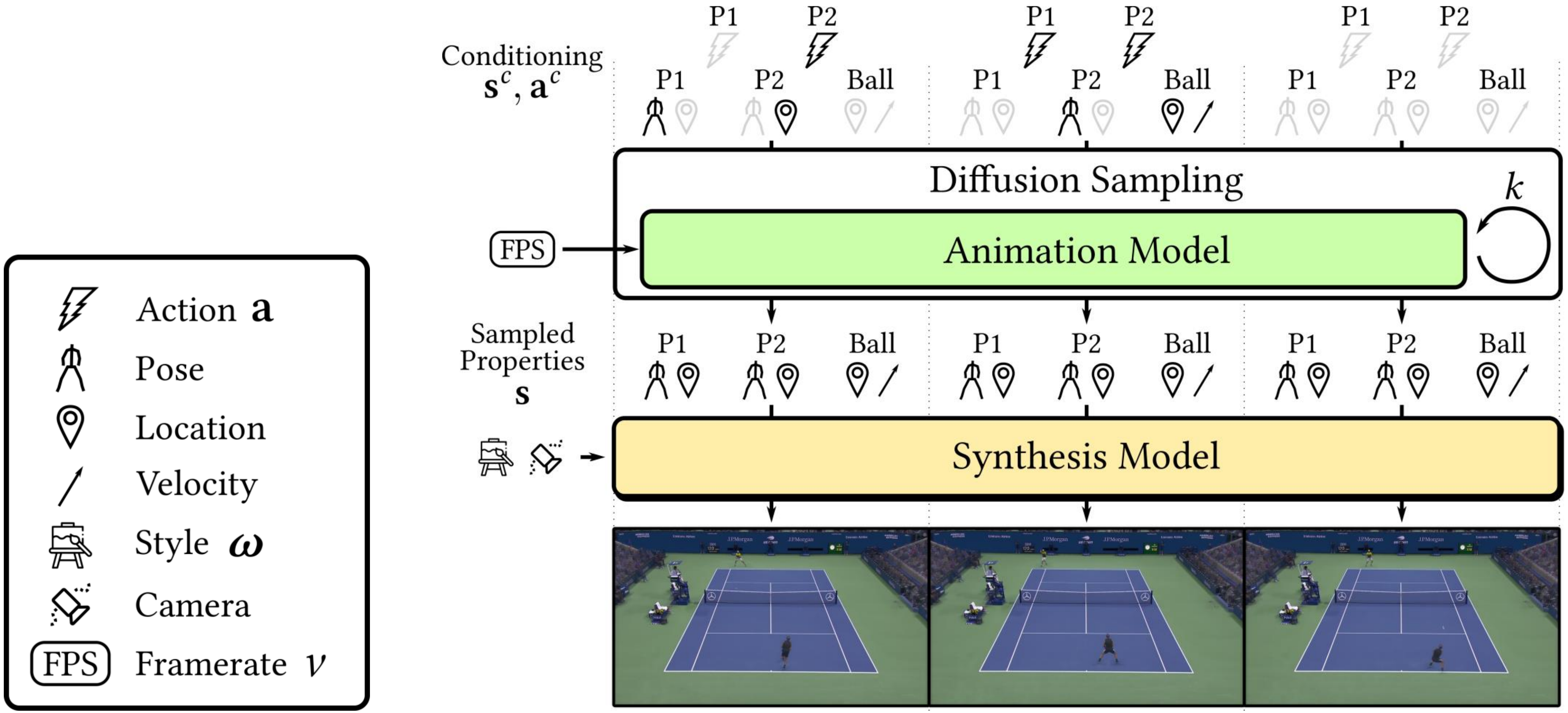
GameGAN
[Kim et. al, CVPR 2020]

Playable Video Generation
[Menapace et. al, CVPR 2021]

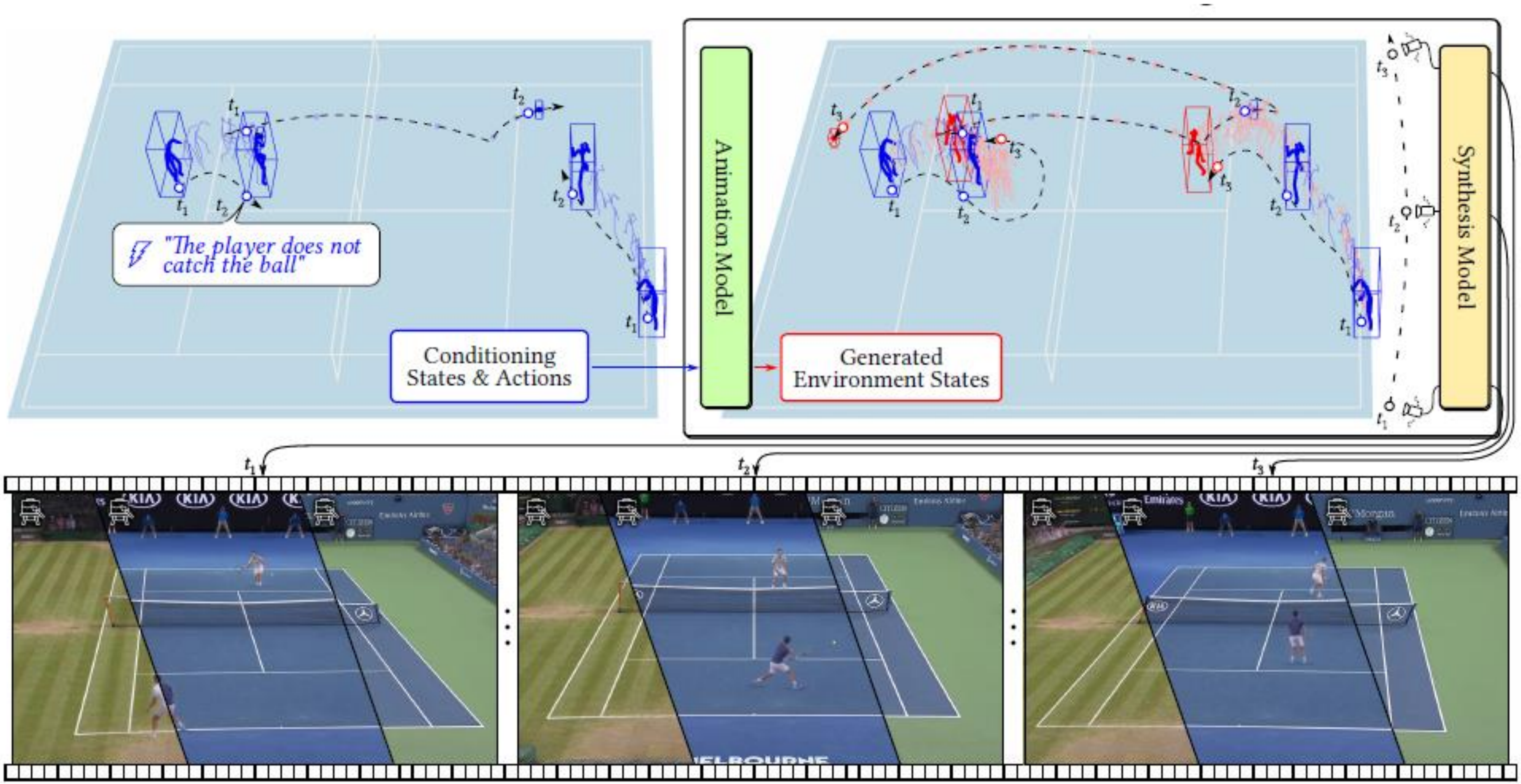
Playable Environments
[Menapace et. al, CVPR 2022]

Method

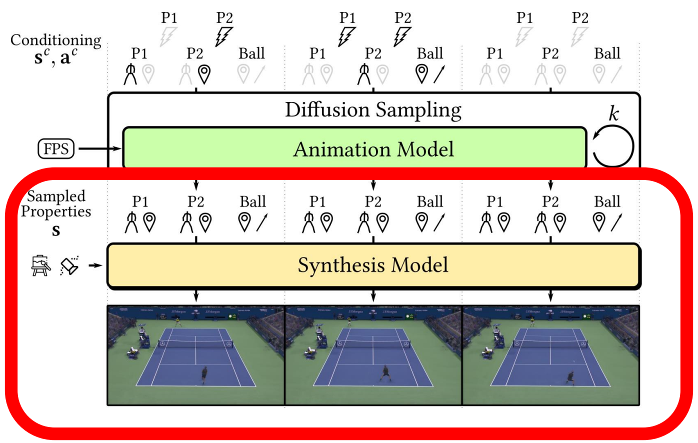
Two separately trained components:



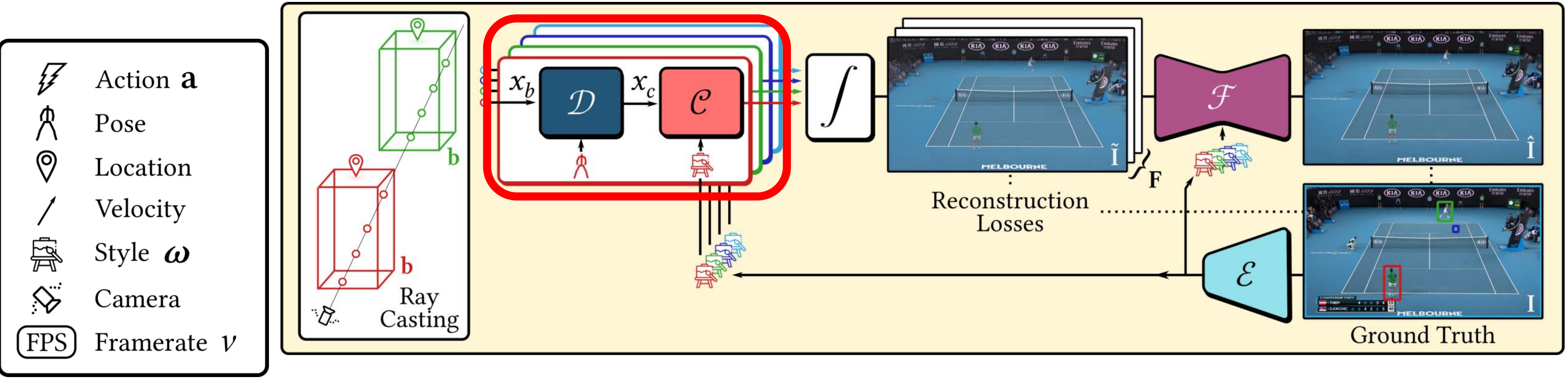
Method


"The player does not catch the ball"
Conditioning States & Actions
Animation Model
Generated Environment States
Synthesis Model

Synthesis Module

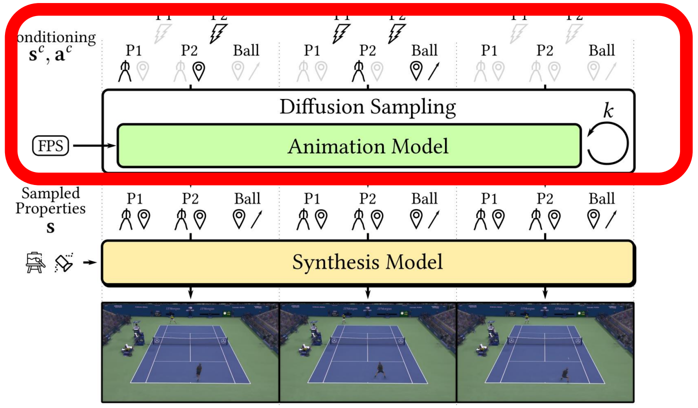


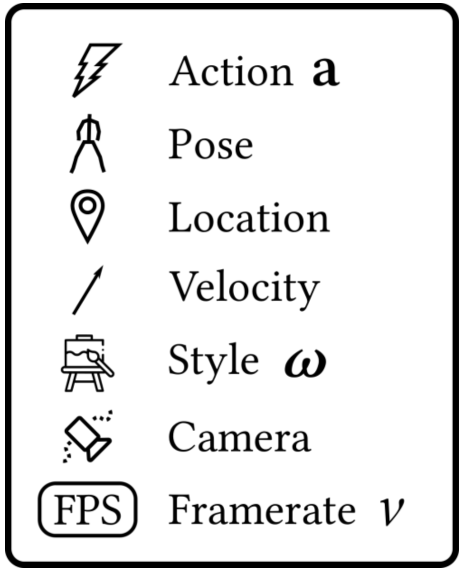
- NERF-based: renders the state of the environment from a given viewpoint
- A composition of NERFS, one for each object
- The model is trained using L2 and perception reconstruction losses

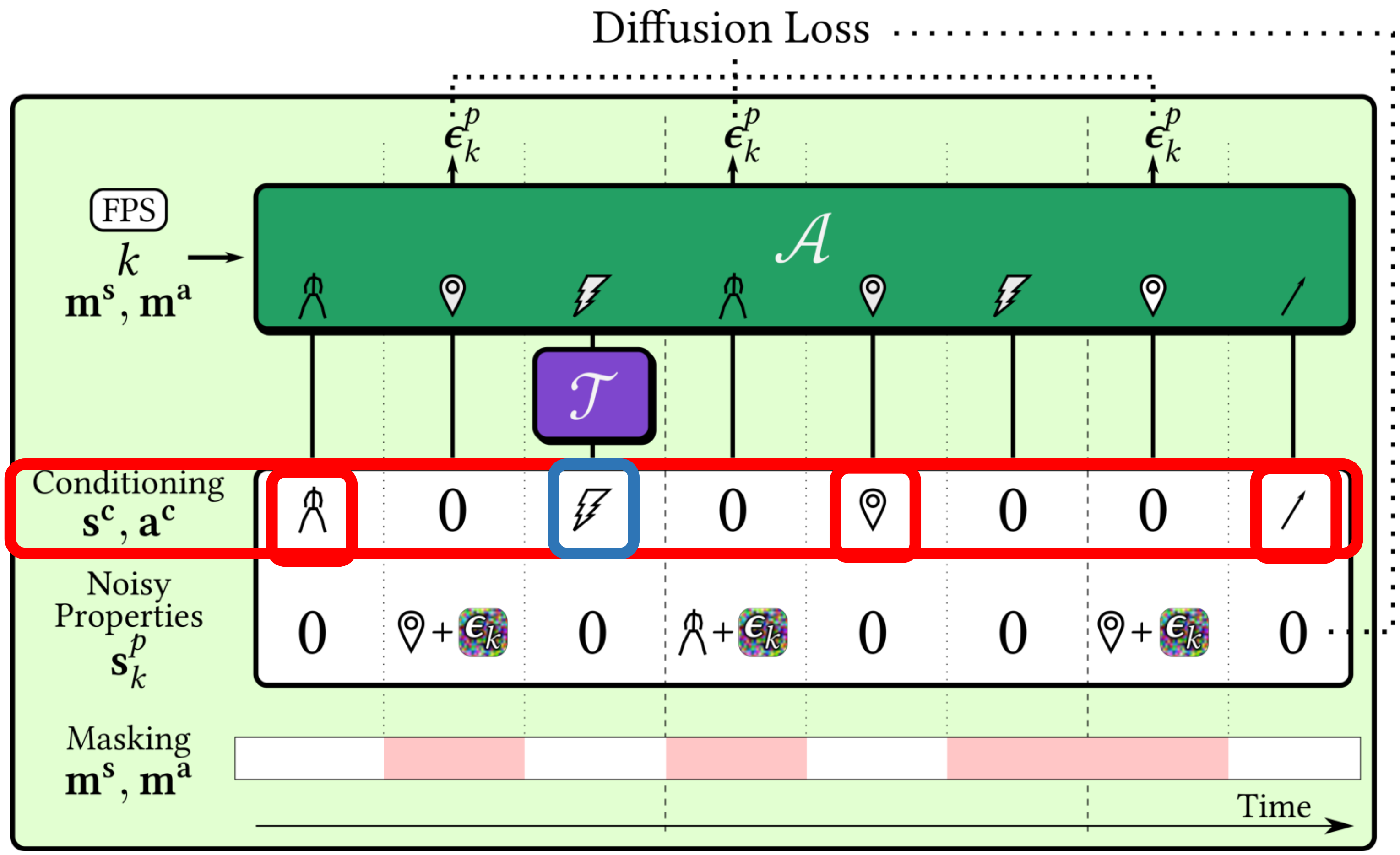


Animation Module

- Diffusion-based: produces sequences of states based on conditioning signals
  - Values: pose, location, velocity of a player or the ball
  - Natural language: what a player is doing

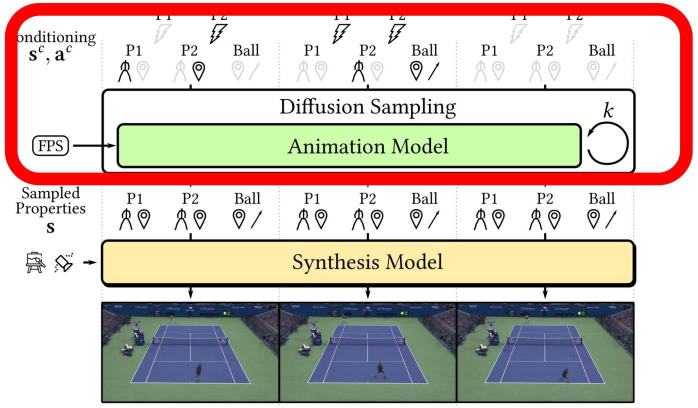


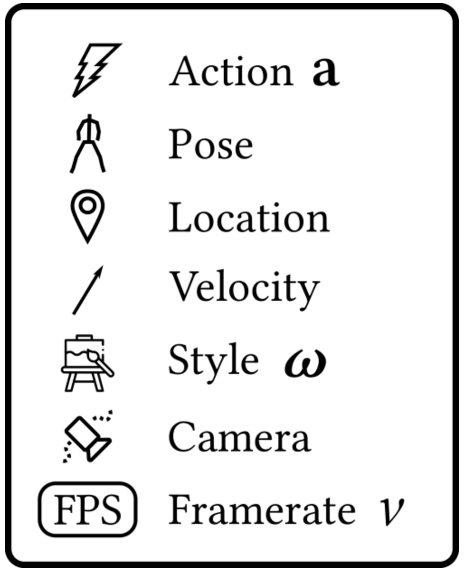


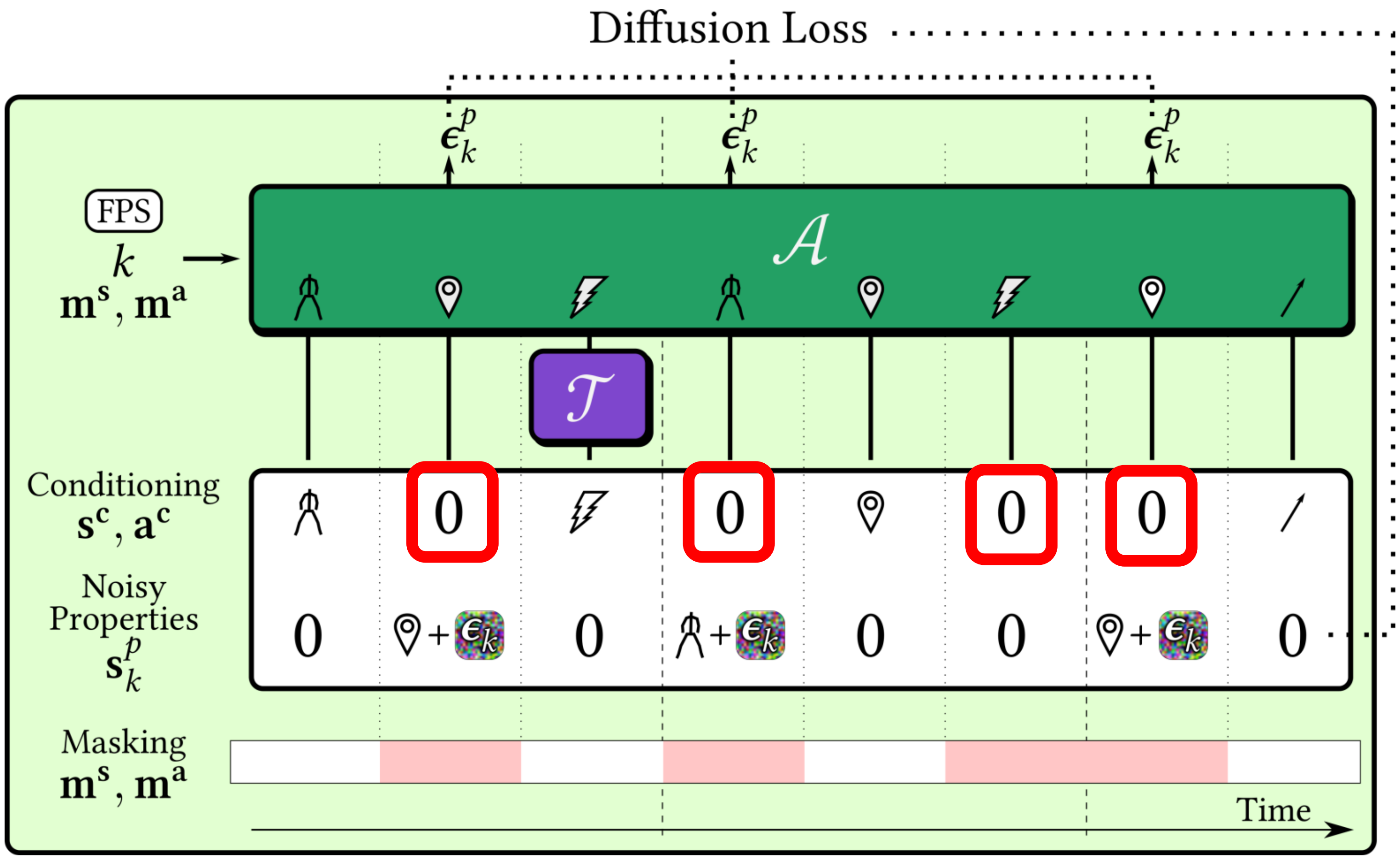


Animation Module

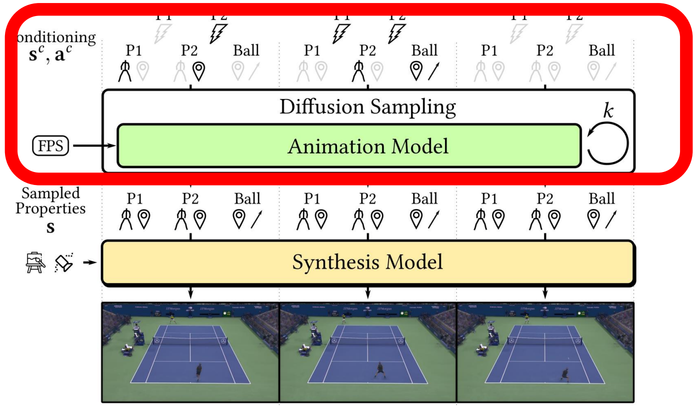
- The conditions are optional: the model can be used at inference time for different task by changing the structure of the conditioning

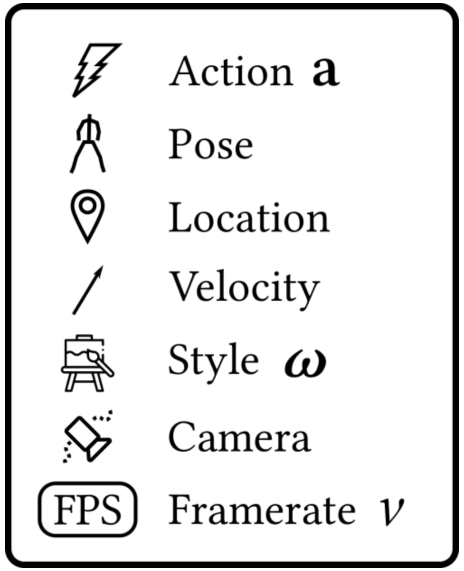




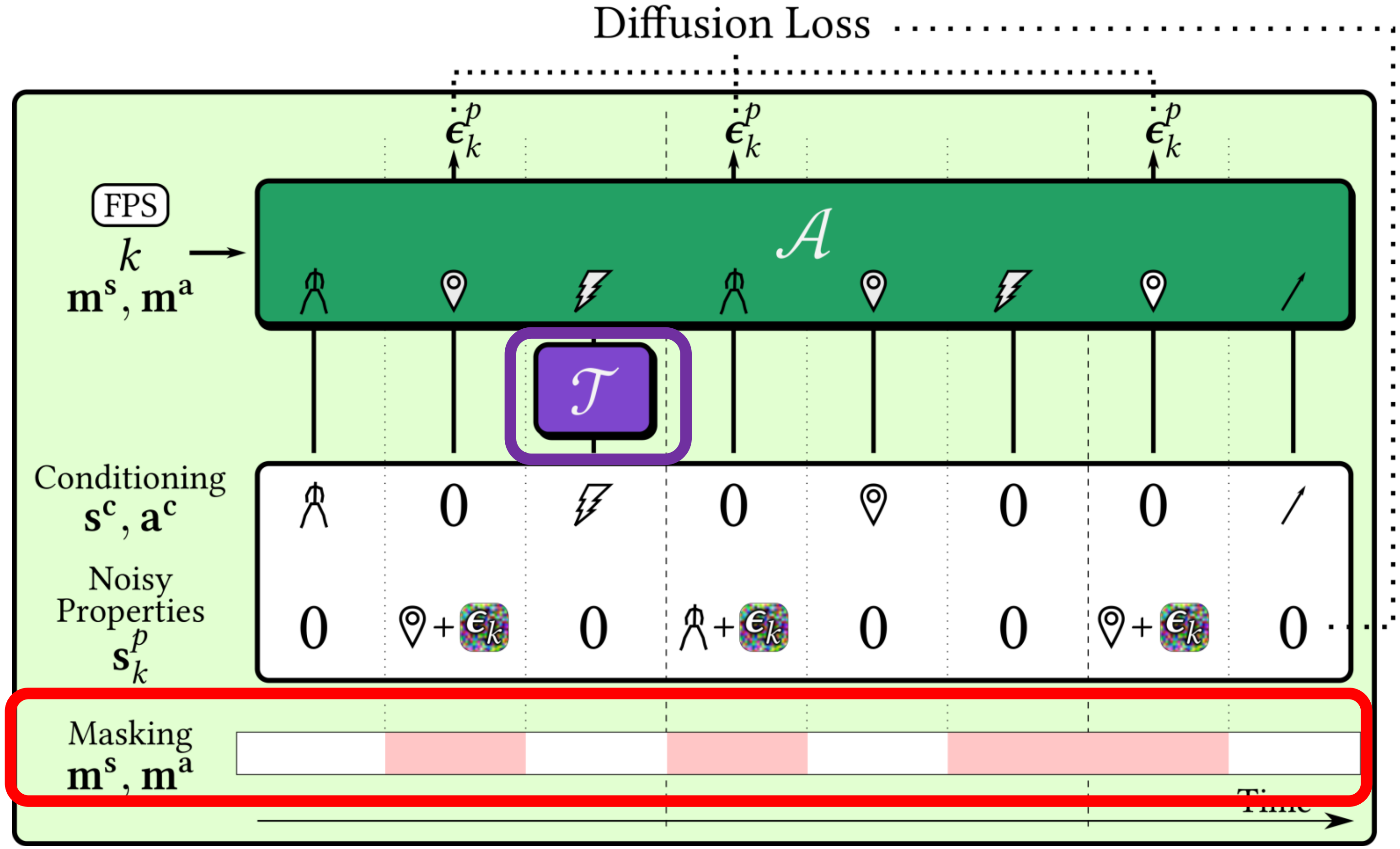


Animation Module

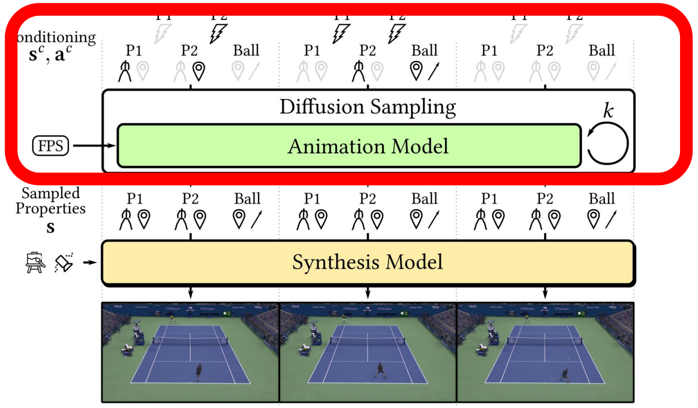


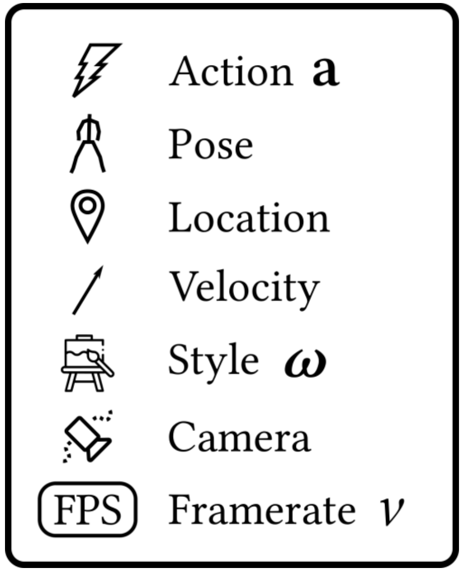


- The model is based on a transformer architecture where a frozen T5 encodes the natural language conditioning
- A mask is specifying which part of the input serves as conditioning and which needs to be predicted

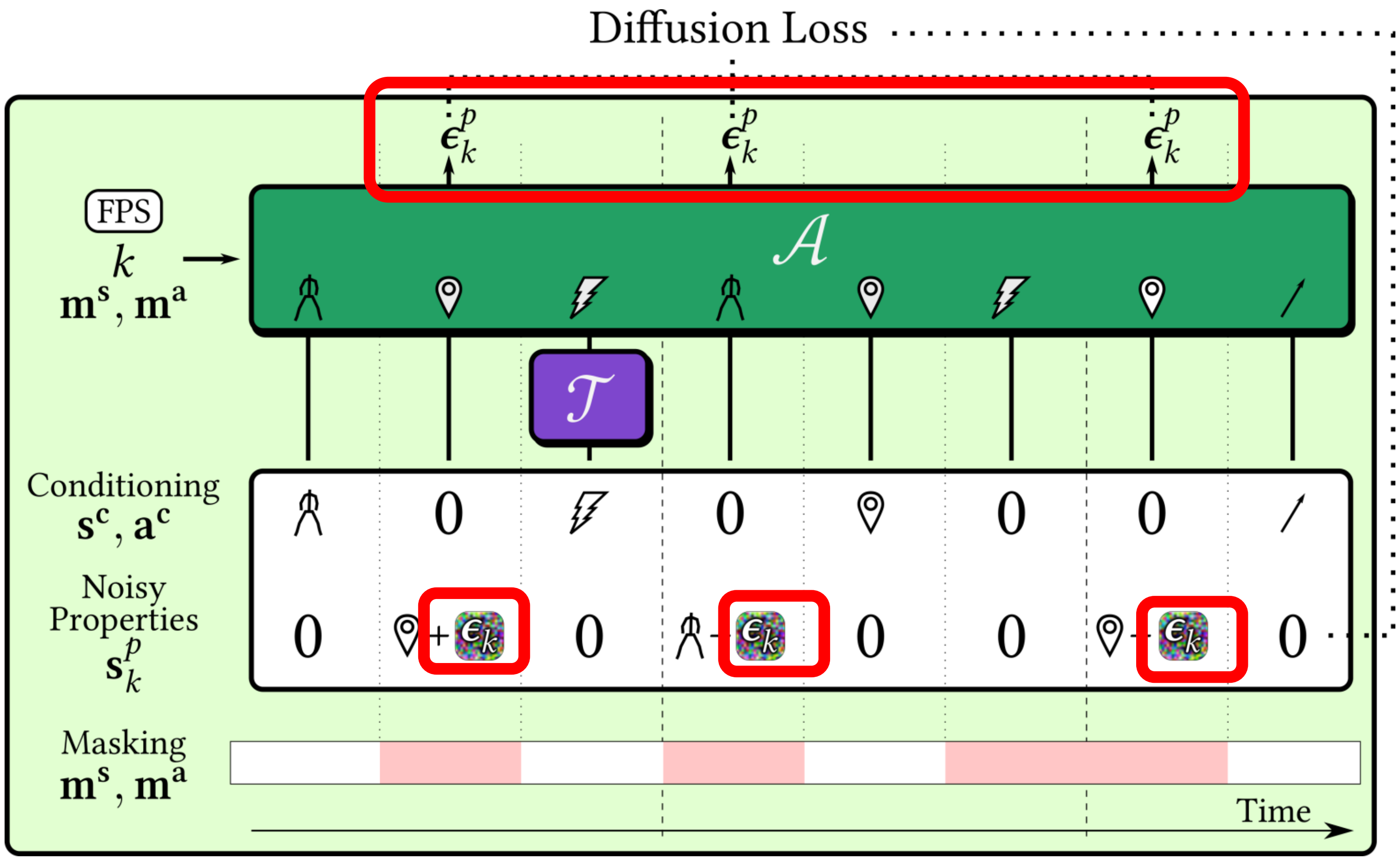


Animation Module

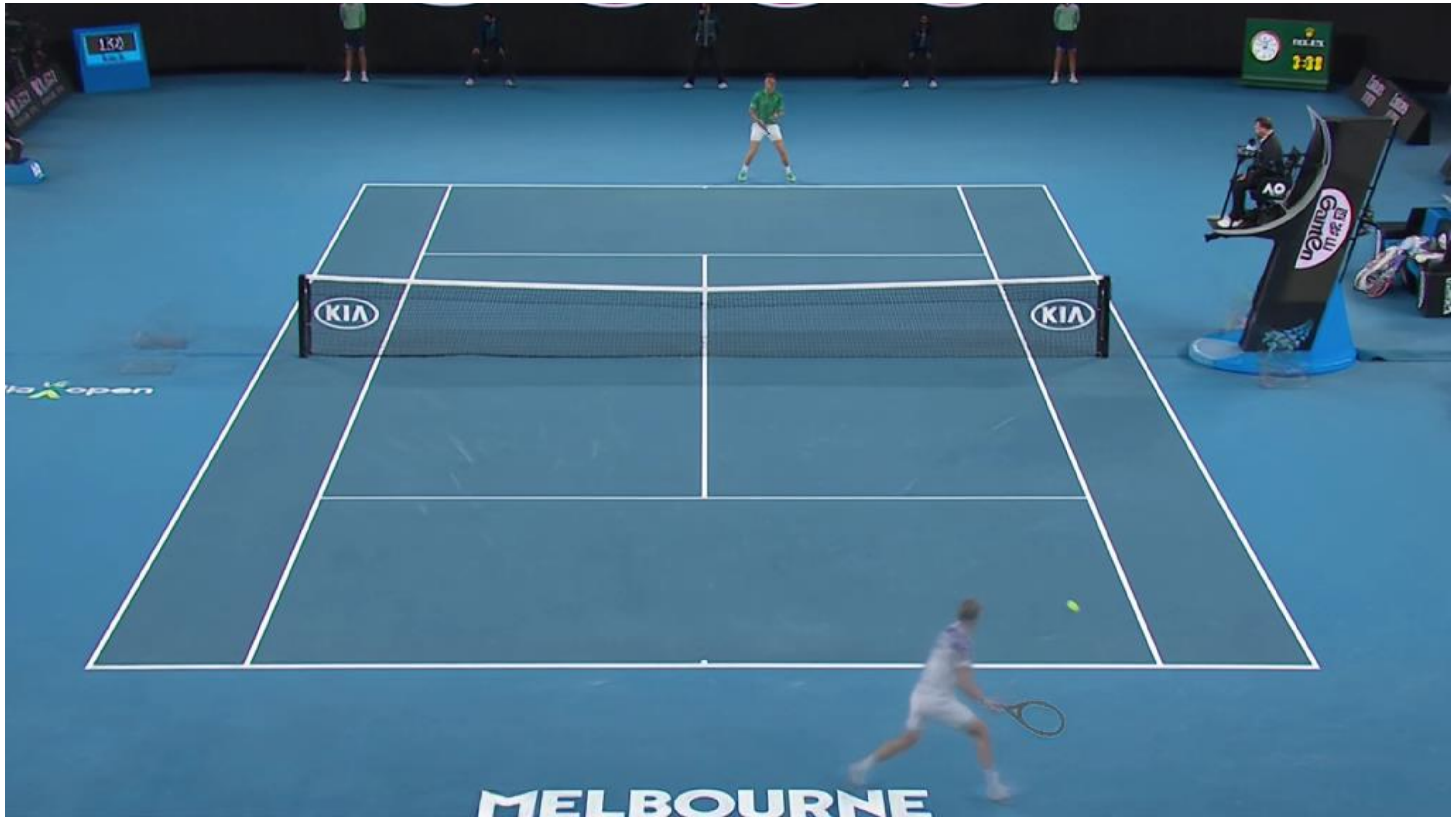




- Finally, the model is trained to predict noise applied to the sequence



Controllable Synthesis

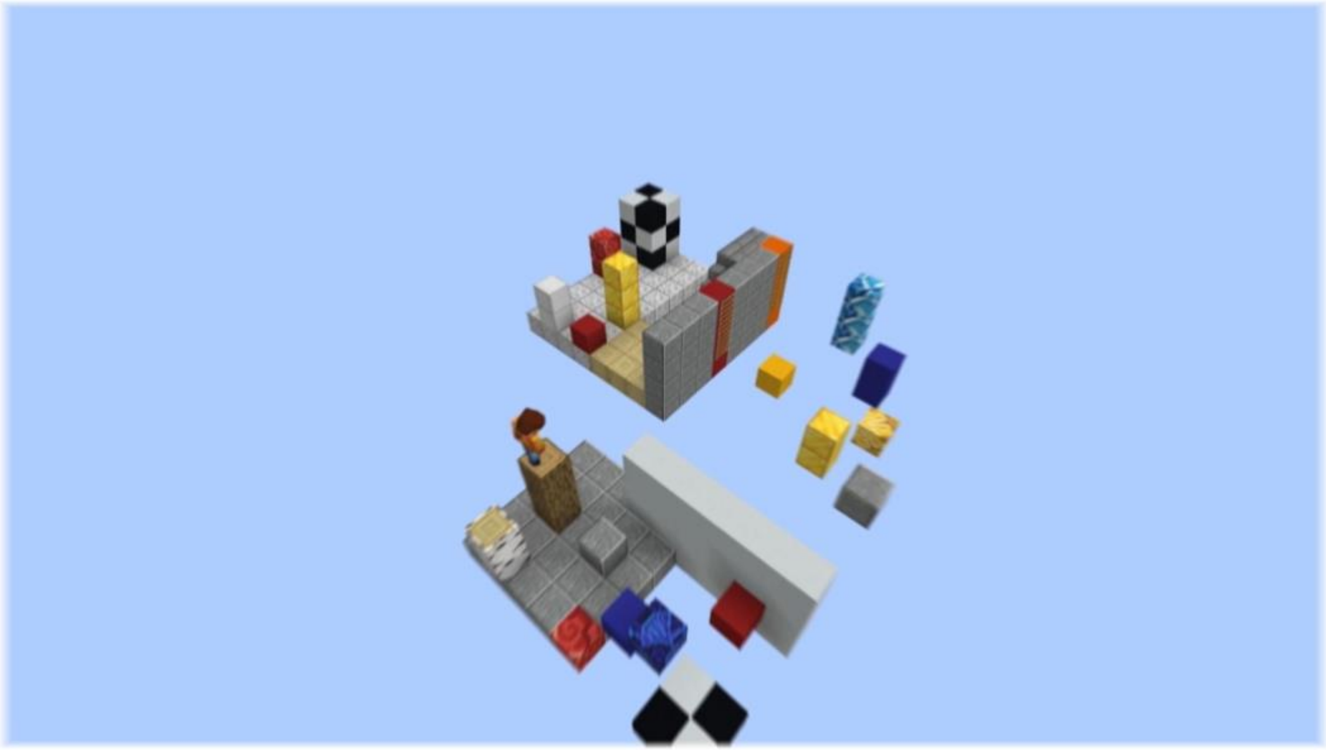


Text-Controllable Animation

Learnable Game Engines:

- Understand physics and game logic
- Can receive action inputs expressed with natural language

Text-Controllable Animation

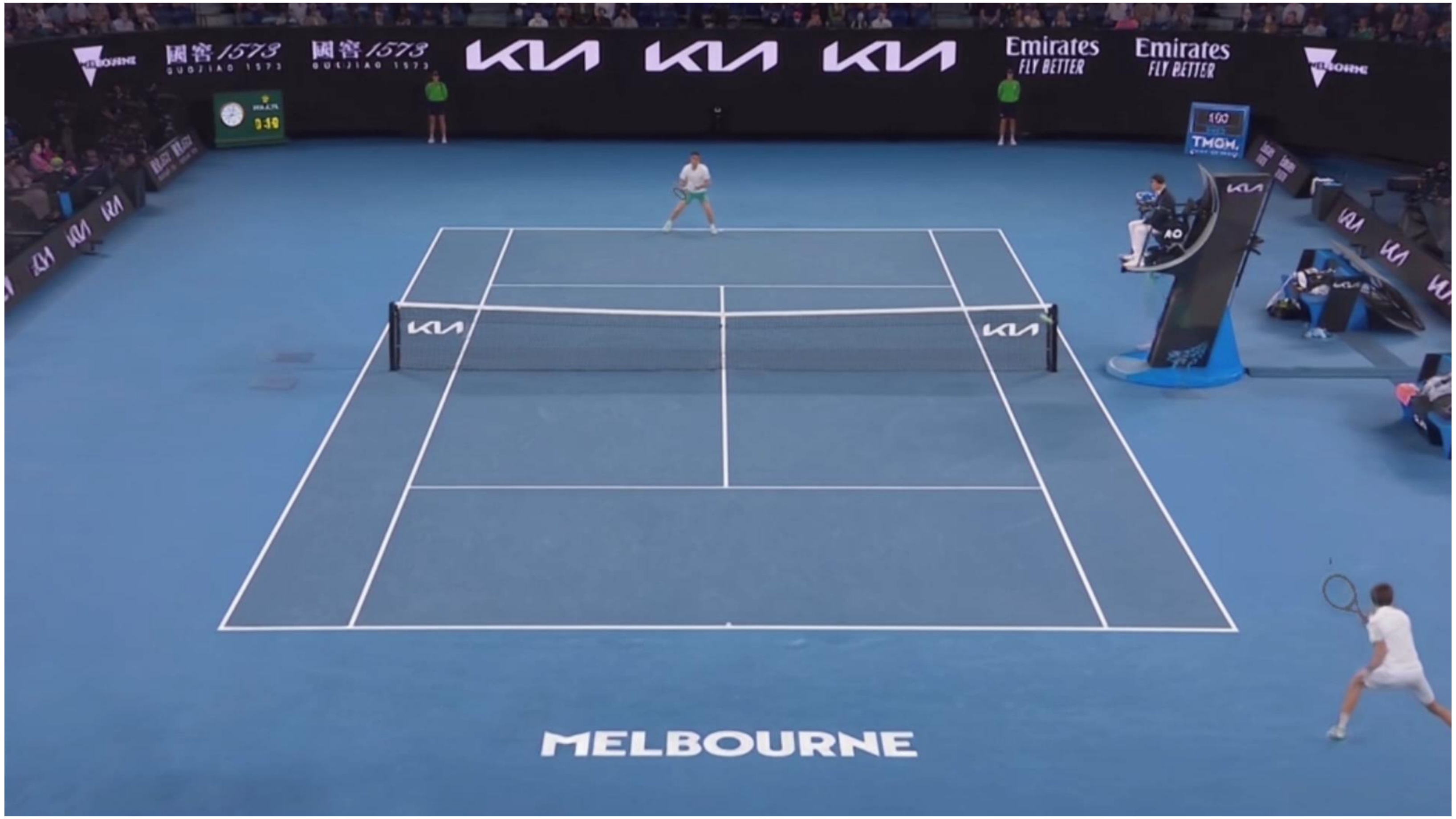


How the player is moving

How the ball is hit

Where the ball is sent

Designing Game Strategy

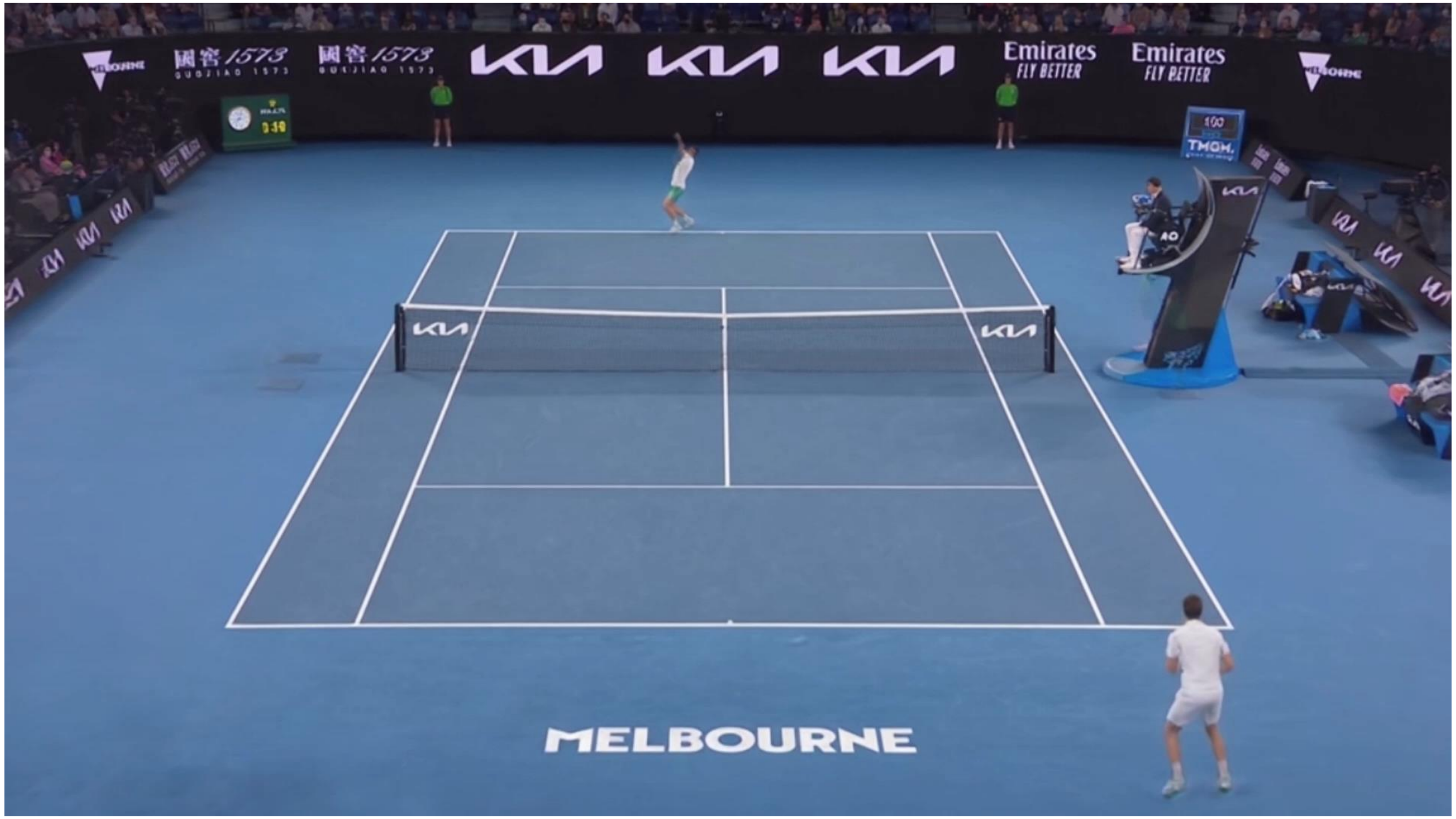


Designing Game Strategy

Making the player win:

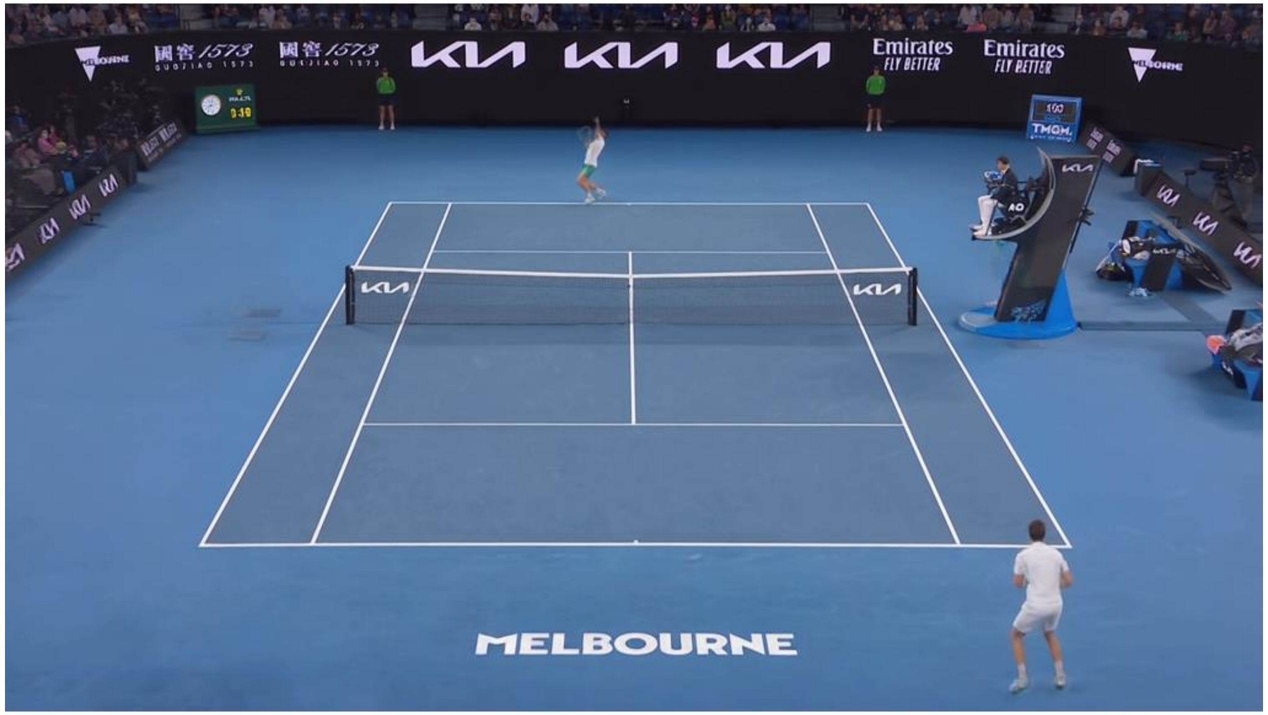
- Reconstruct the scene
- Devise winning actions
- Animate players
- Render the results

Designing Game Strategy



Designing Game Strategy

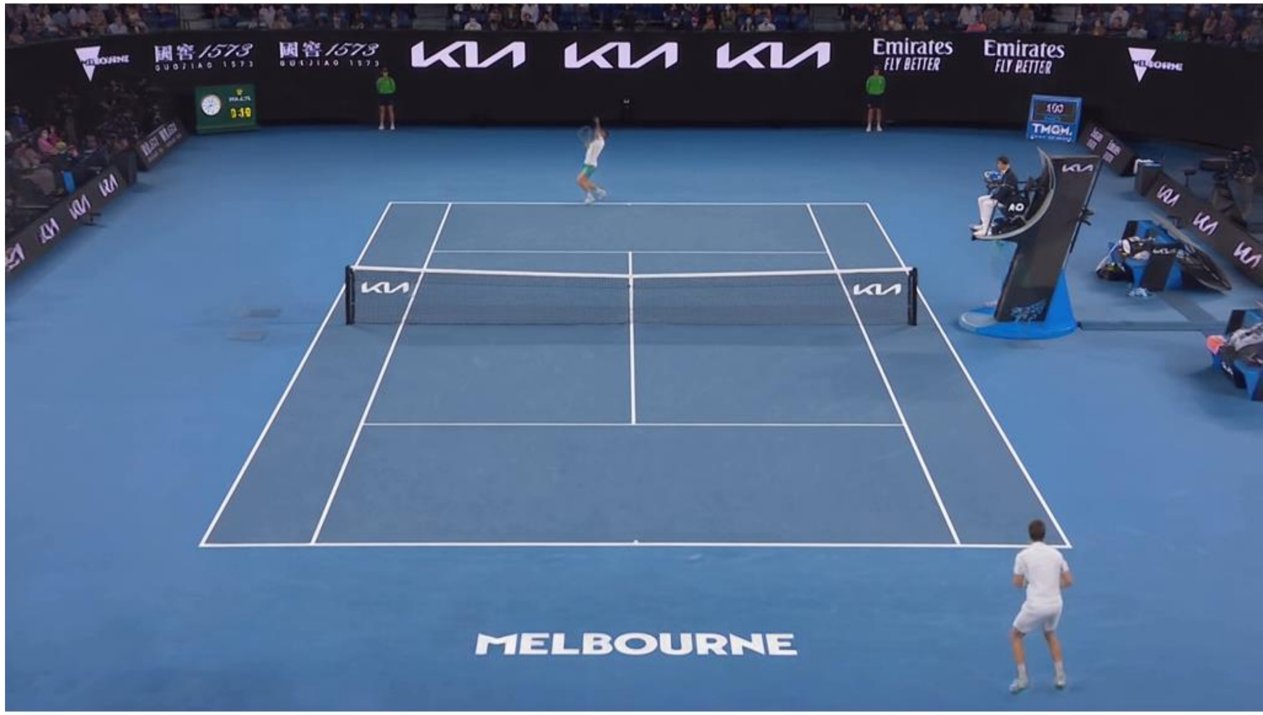
the player serves and sends the ball to the right service box



The player stands still waiting for a serve

Original video = Bottom player loses

the player serves and sends the ball to the right service box



The player stands still waiting for a serve

½ Original video + “The [TOP] player doesn’t catch the ball”= Bottom player wins

Play LGEs as Videogames

Play Against an Opponent

LGE Opponent Control

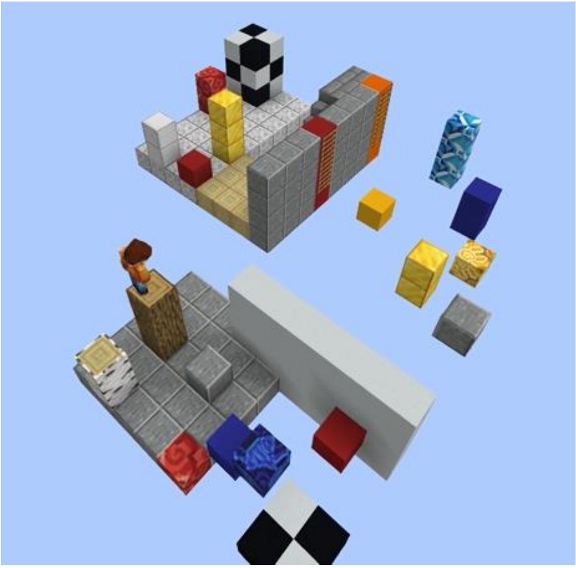
Director’s Mode

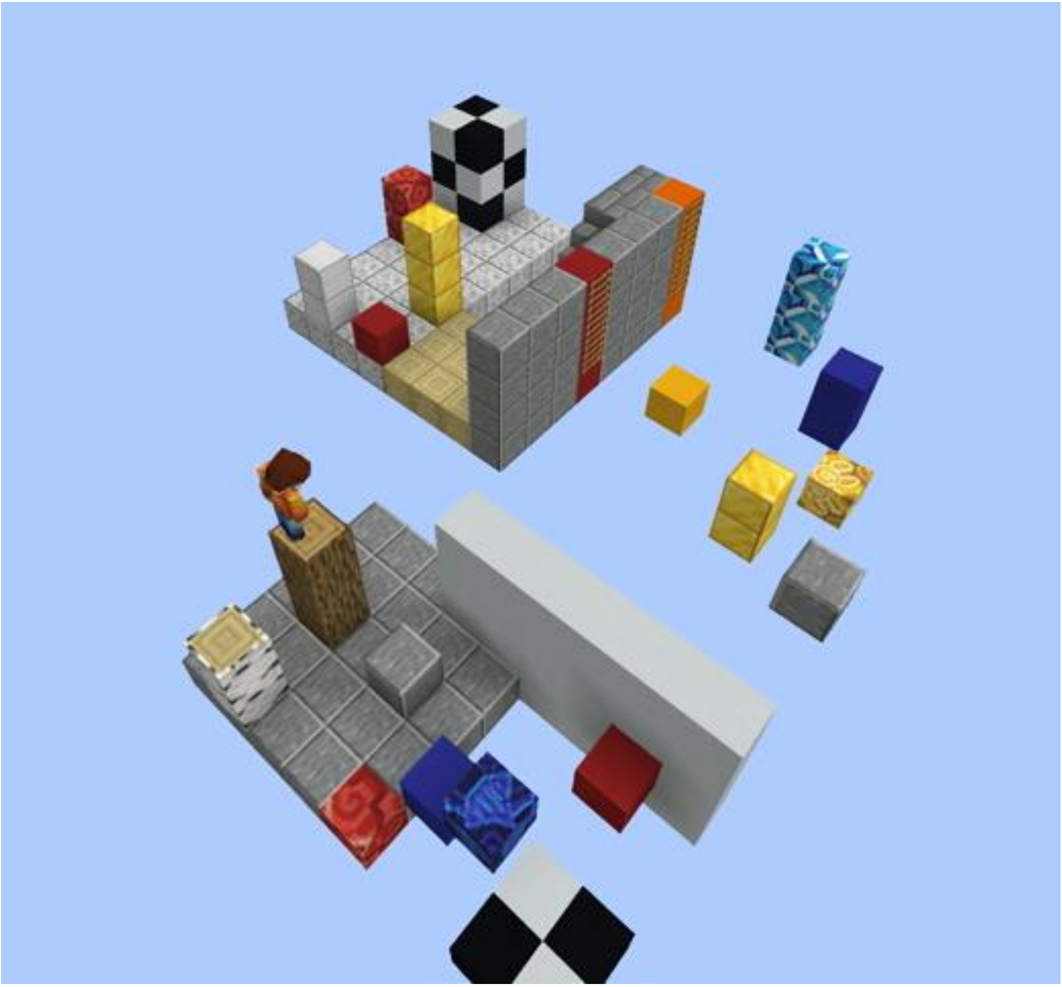
Constrain generation using:

- Desired values of the environment states
- Actions expressed with natural language

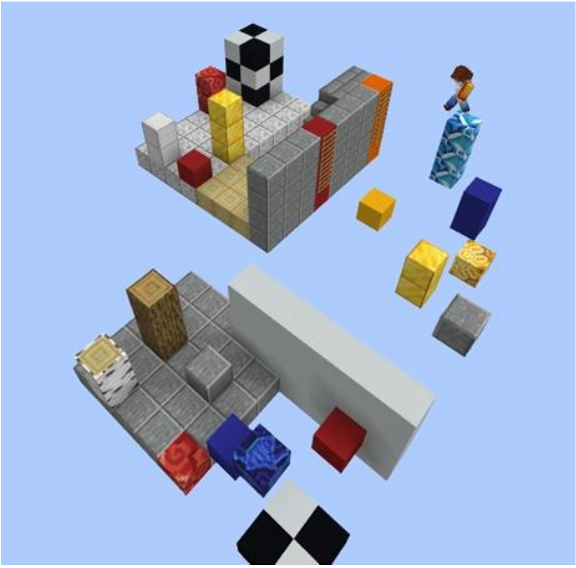
Director’s Mode

First Frame

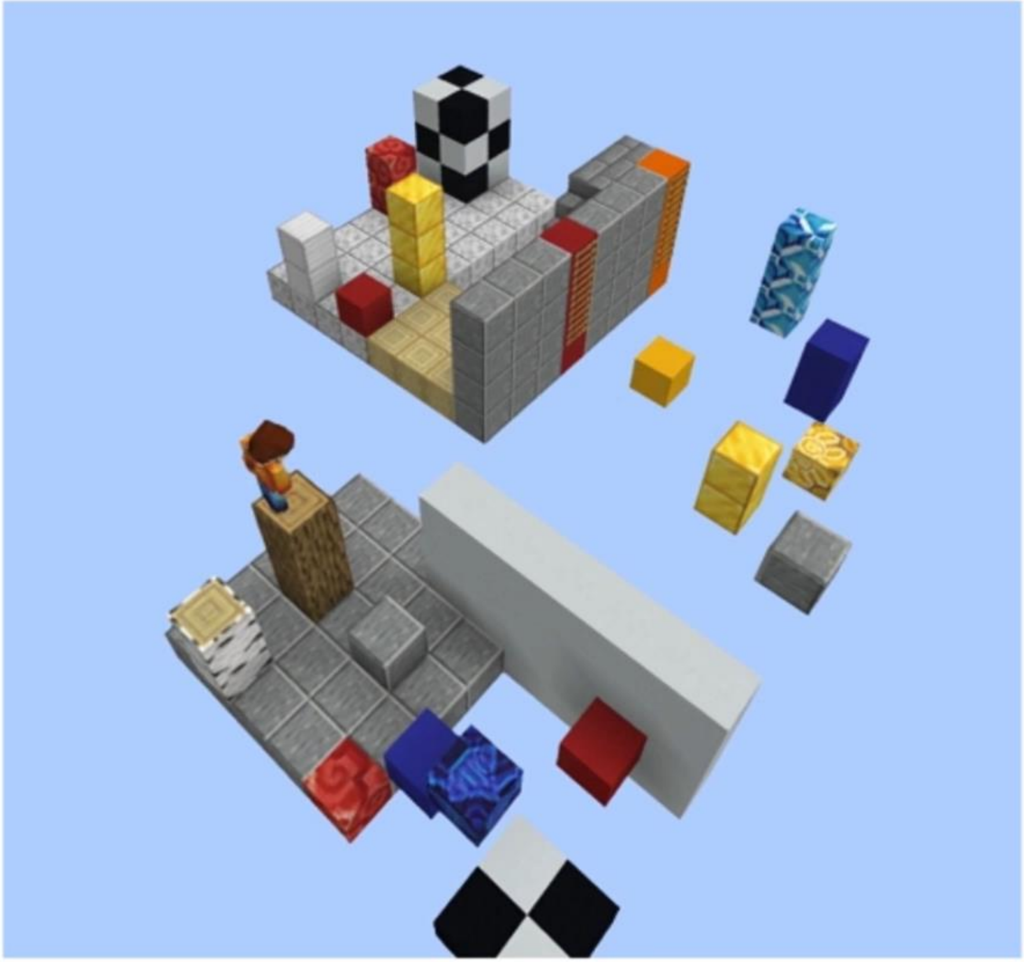


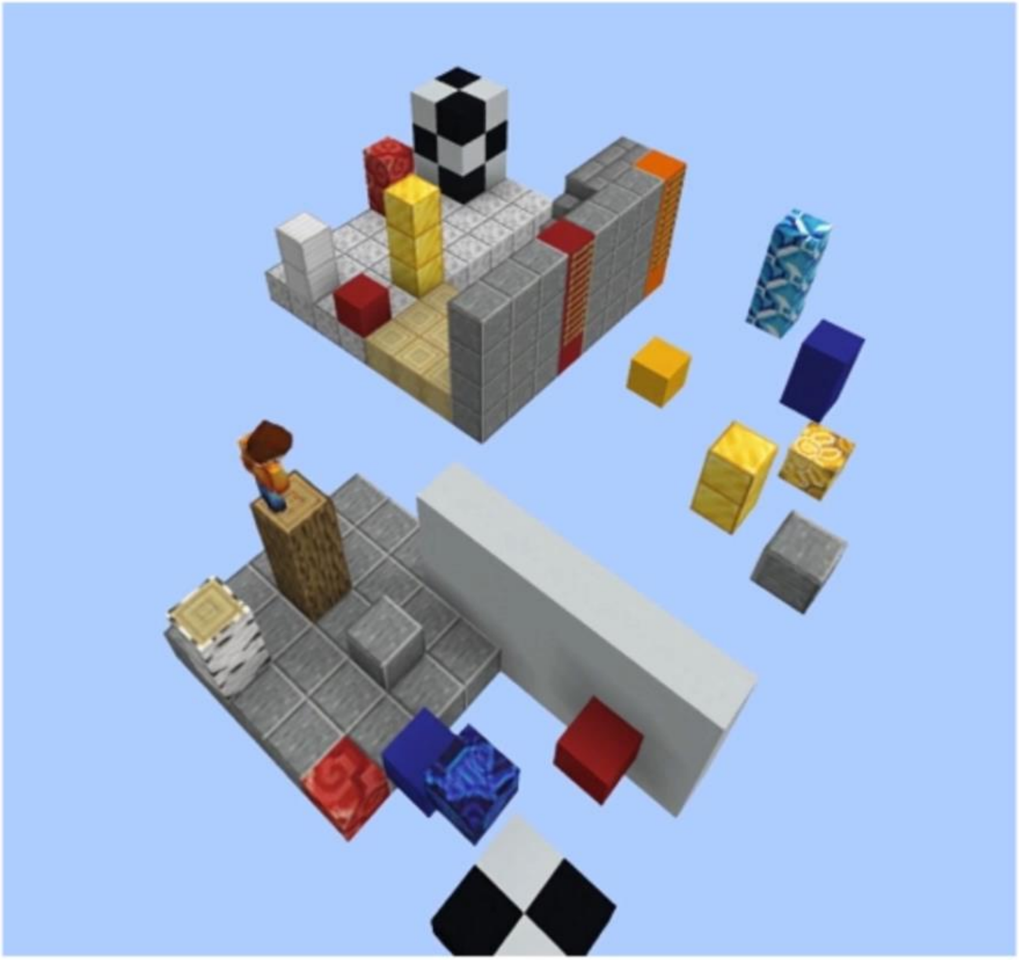


Last Frame



Director’s Mode

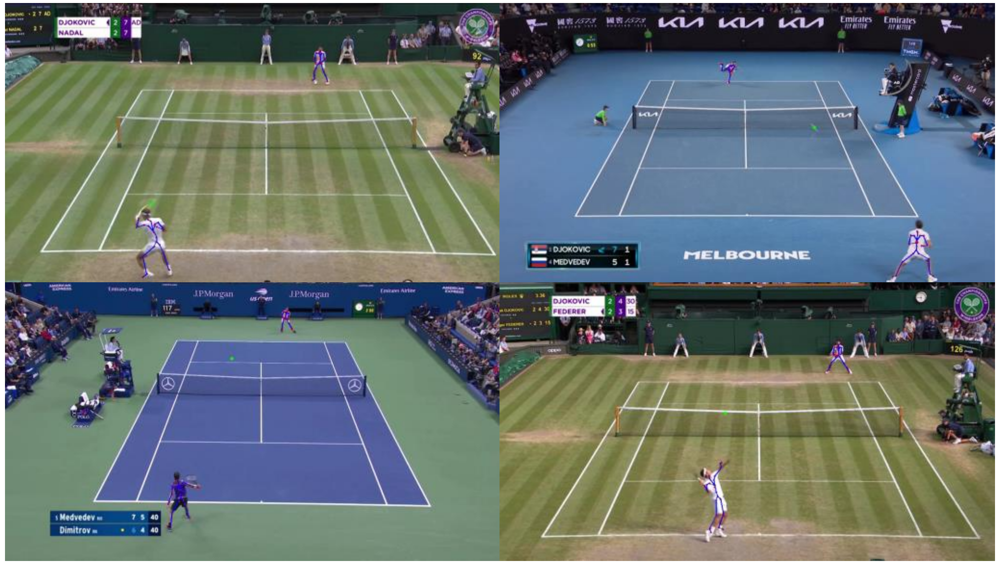




Director’s Mode

The conditioning is flexible, e.g., give multiple actions to constrain the solution

LGE Datasets



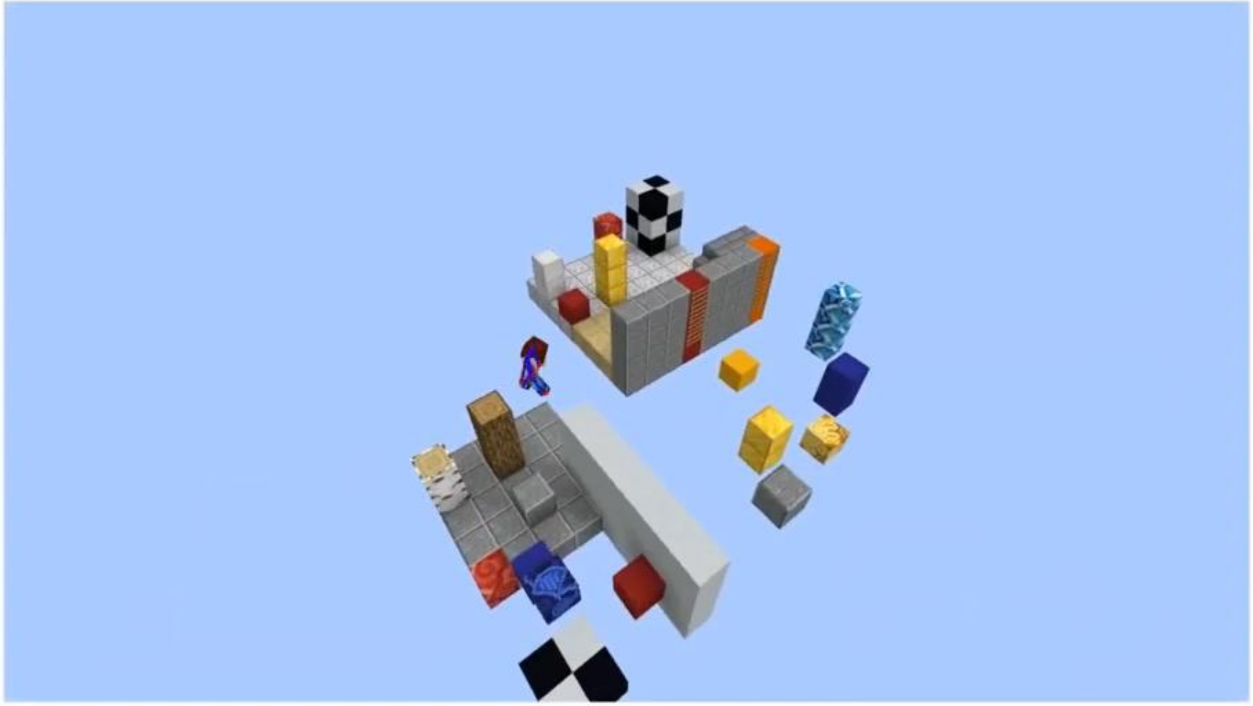
Tennis

- 7112 video sequences at 1920x1080@25fps
- 15.5 hours of videos
- 1.12M fully annotated frames
- 25.5k unique captions

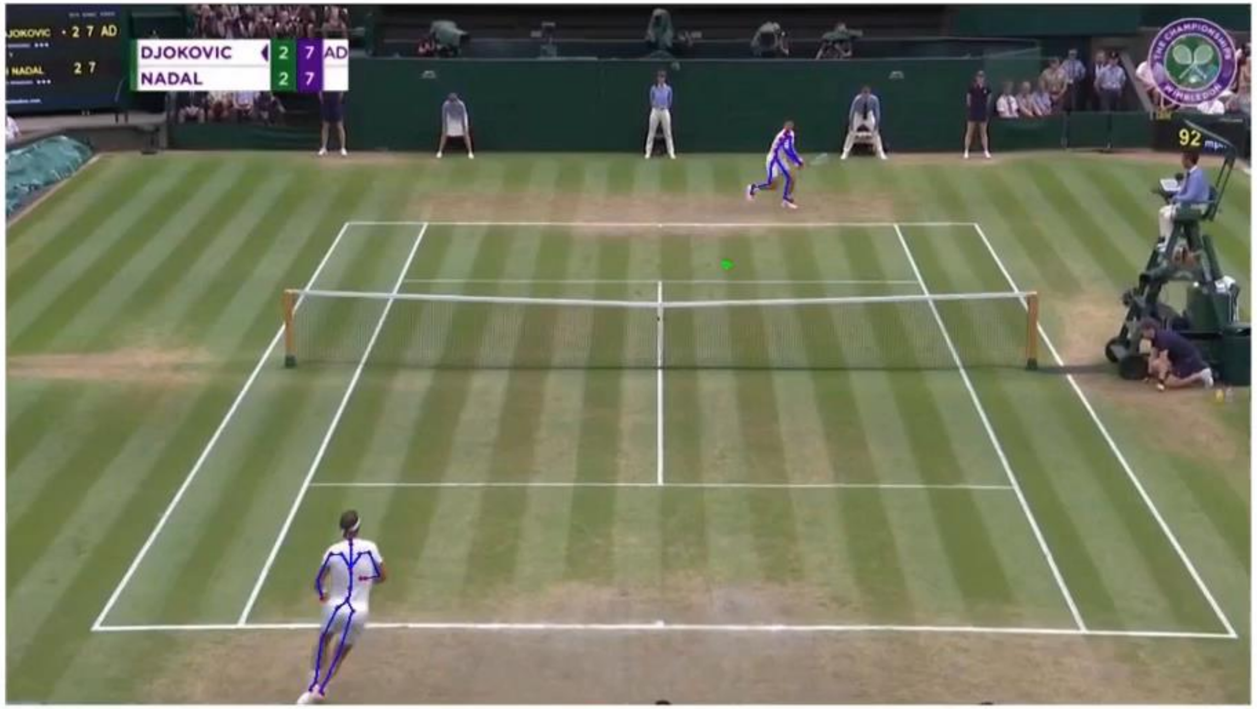
Minecraft

- 61 video sequences at 1024x567@20fps
- 1.2 hours of videos
- 68.5k fully annotated frames
- 1.24k unique captions

LGE Datasets

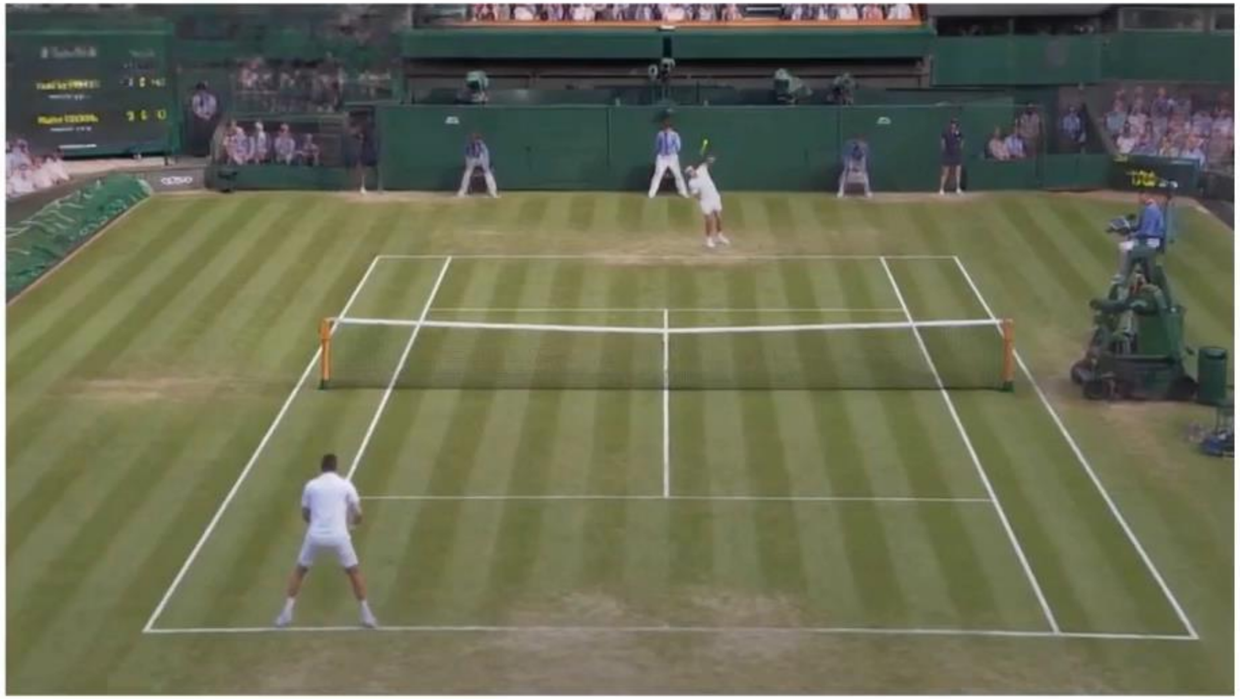


Minecraft

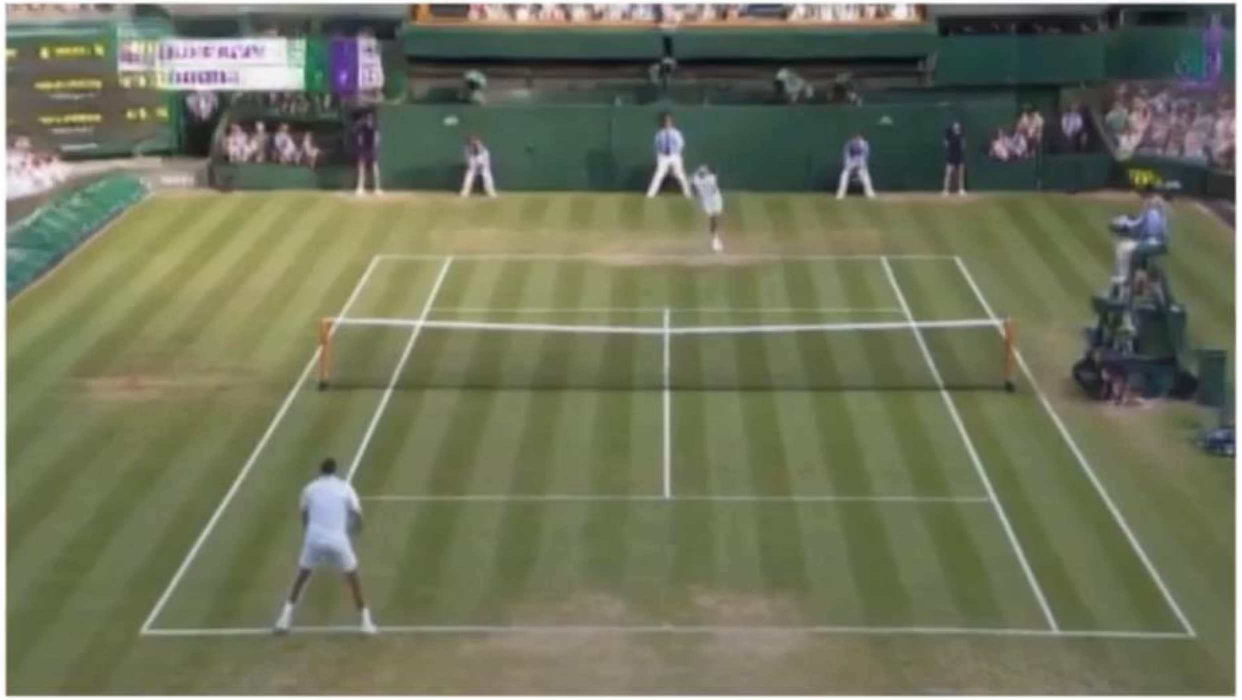


Tennis

Synthesis Model Evaluation



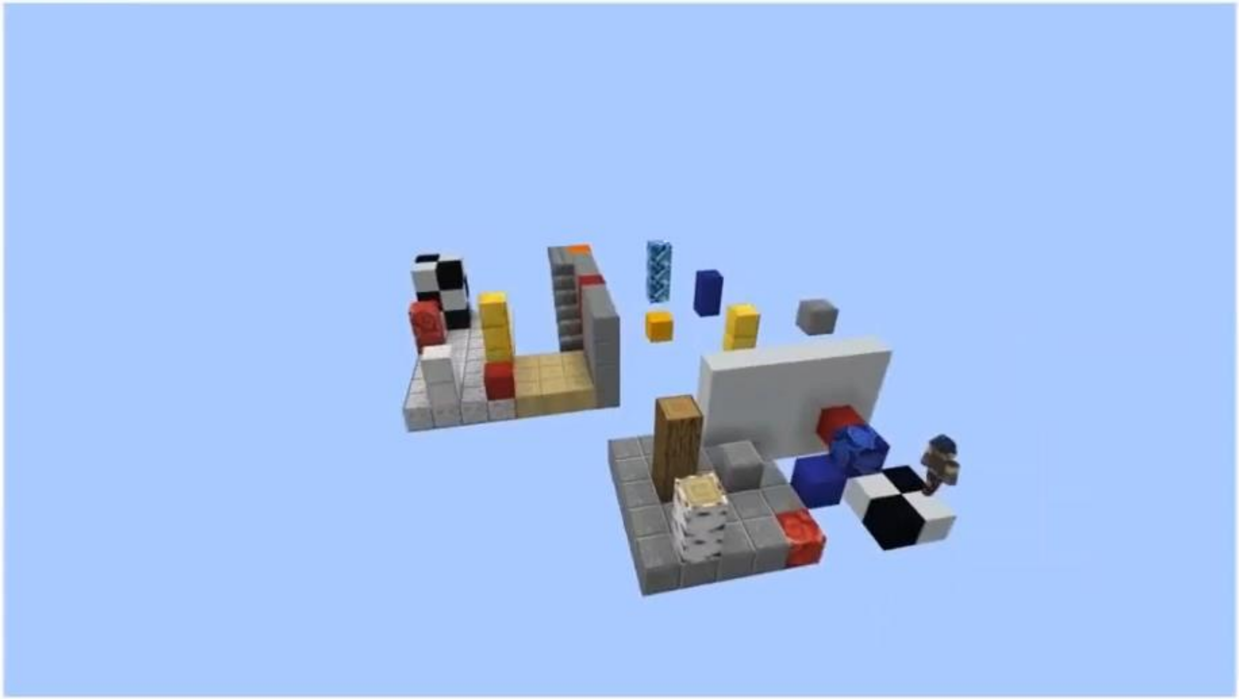
Learnable Game Engines



Playable Environments

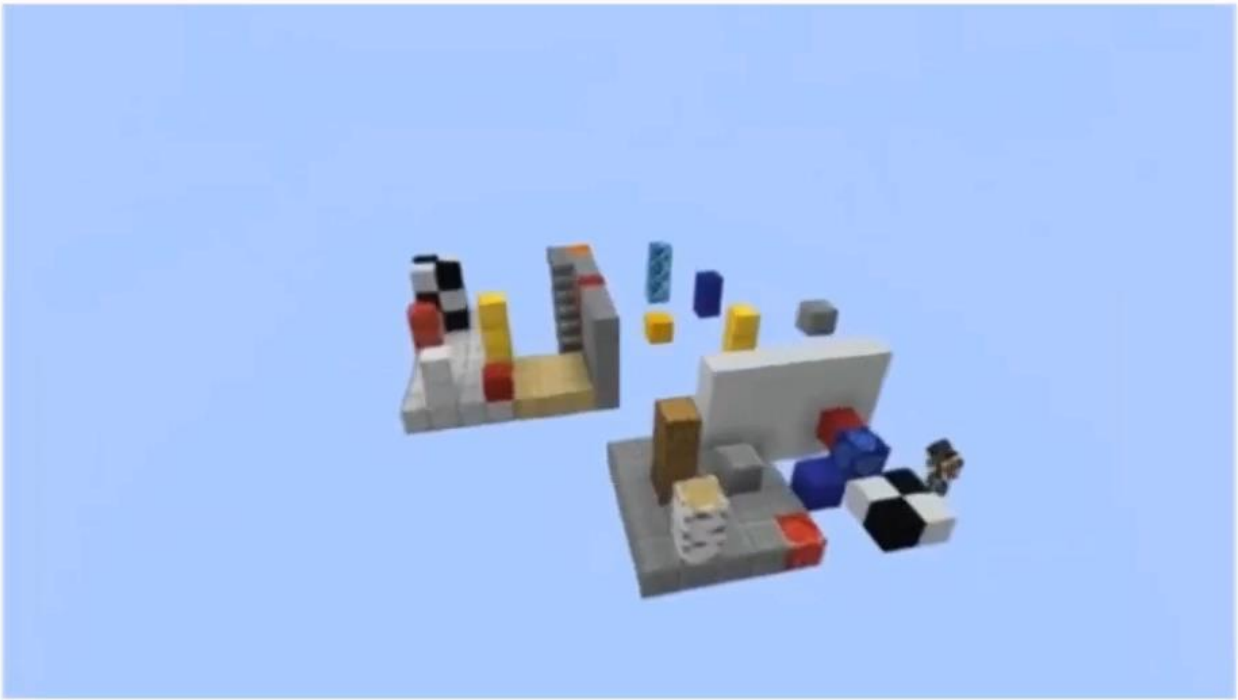
- Increased resolution
- No checkerboard artifacts

Synthesis Model Evaluation



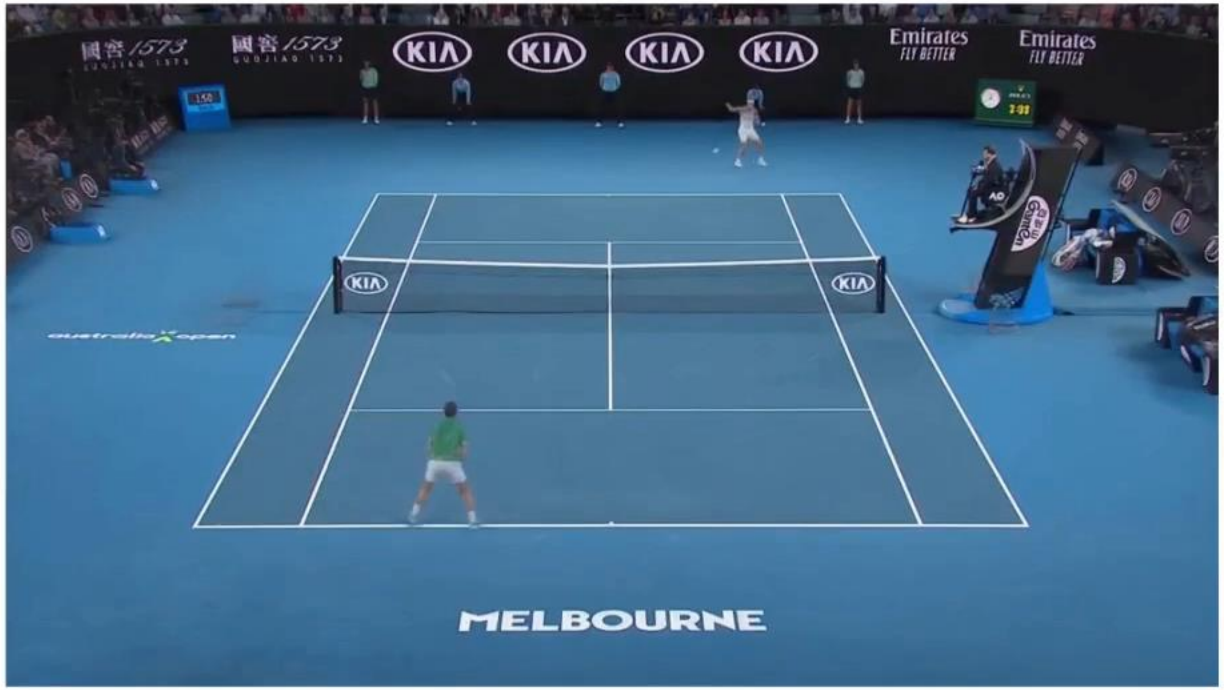
Learnable Game Engines

- Increased resolution
- No checkerboard artifacts



Playable Environments

Animation Model Evaluation



Learnable Game Engines

Playable Environments

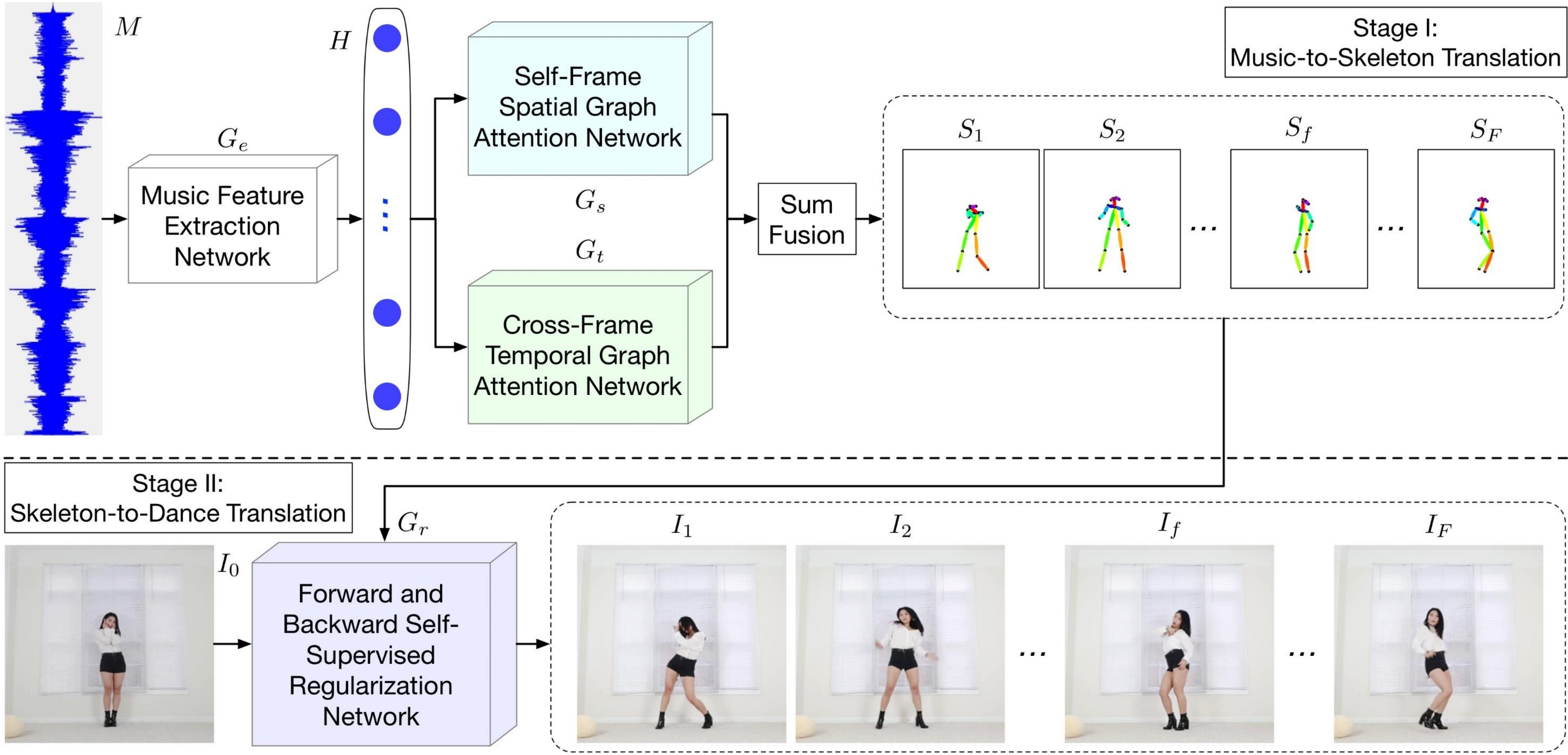
- Higher quality and higher frame rate sequences
- Better scene dynamics

Beyond Playable Environments

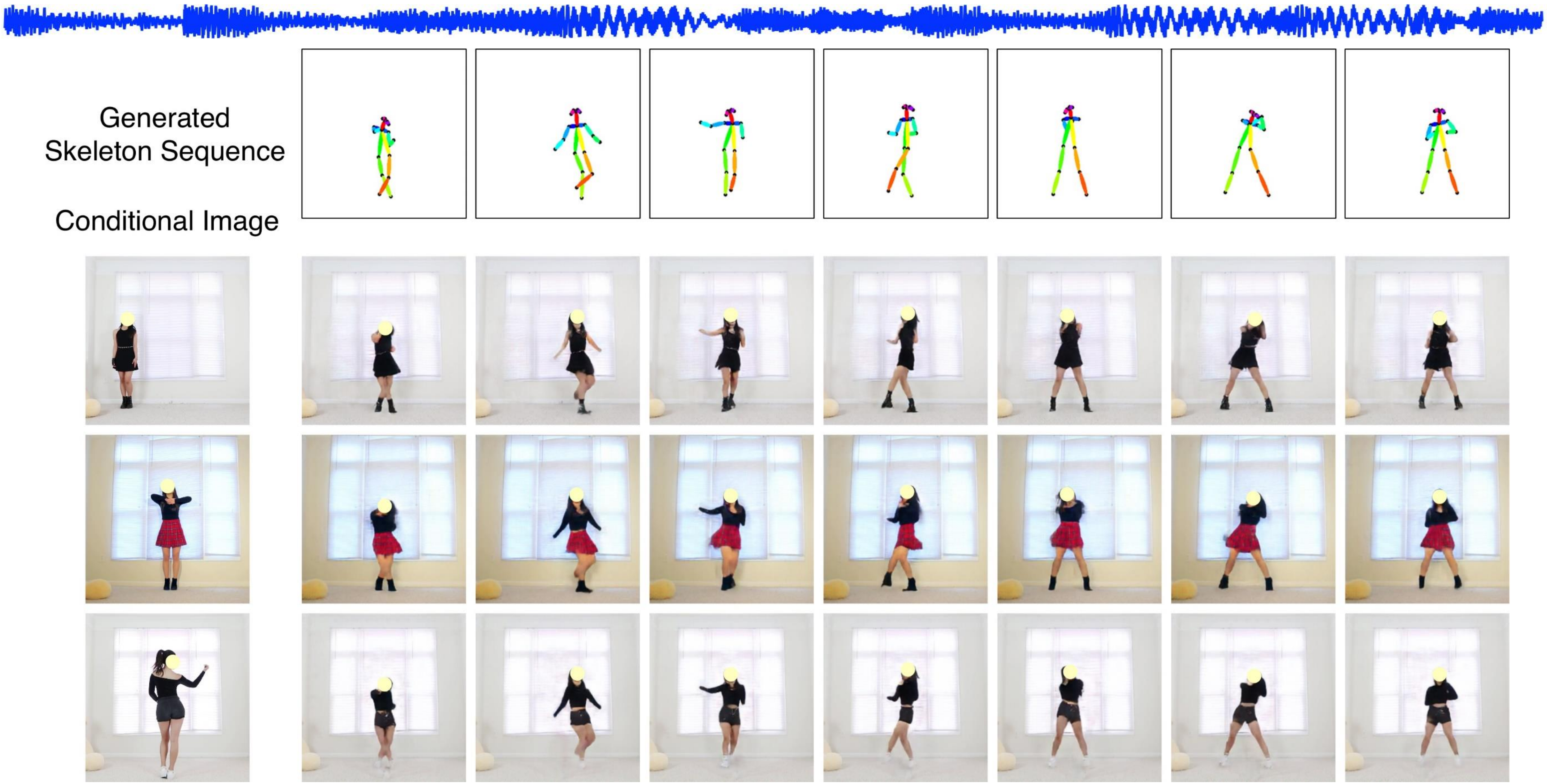
- Can we generate large scenes with manipulable objects inside?
- Can we do that without object localization and camera calibration?
- This environment representation can be used to model complex games with many objects and large environment

Music-Guided Dance Video Synthesis

DanceGAN


M
G_e
Music Feature Extraction Network
H
Self-Frame Spatial Graph Attention Network
G_s
G_t
Cross-Frame Temporal Graph Attention Network
Sum Fusion
Stage I: Music-to-Skeleton Translation
S_1
S_2
S_f
S_F
Stage II: Skeleton-to-Dance Translation
G_r
I_0
Forward and Backward Self-Supervised Regularization Network
I_1
I_2
I_f
I_F

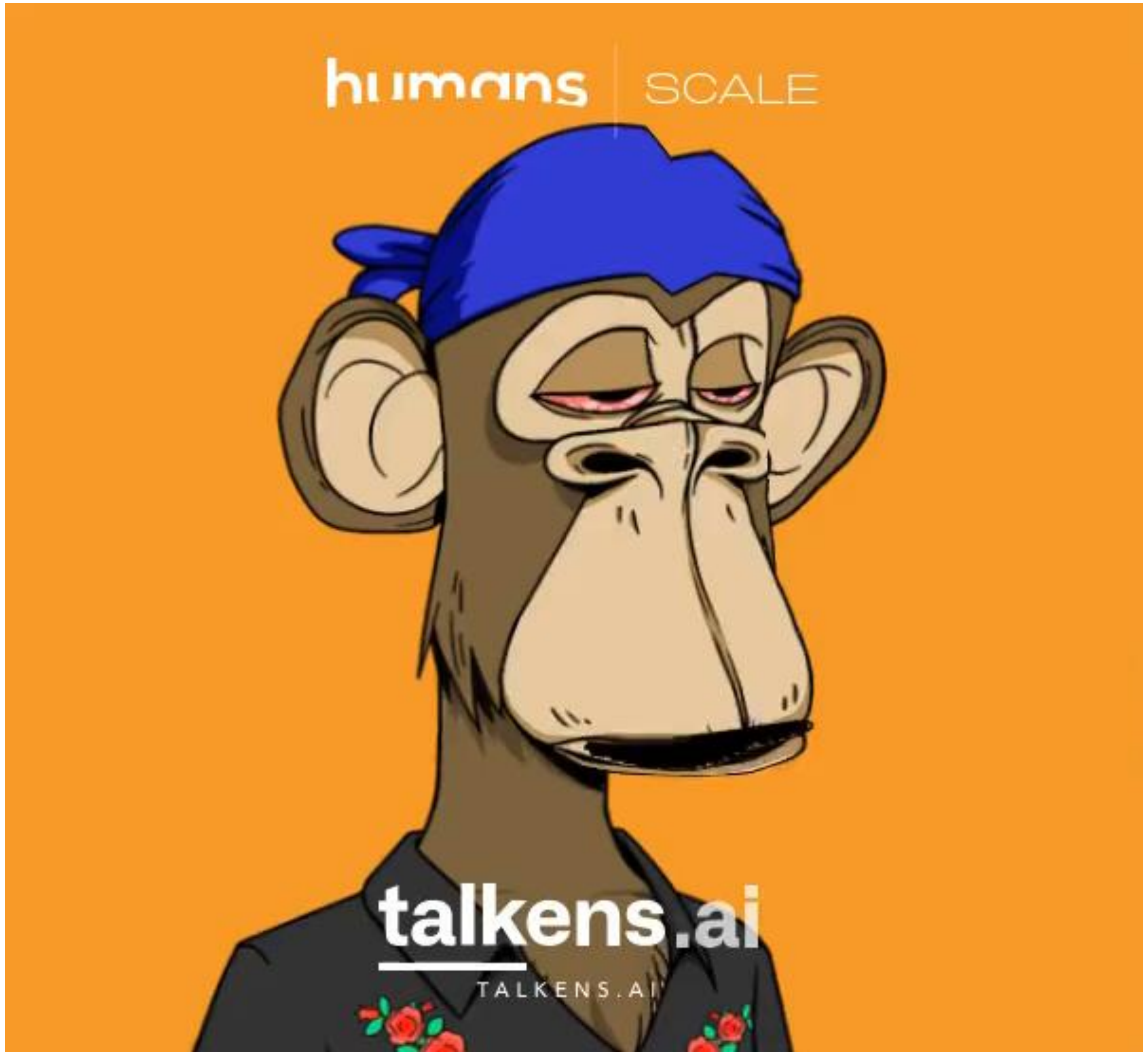
Music-Guided Dance Video Synthesis



Demo

Where Are We Going Now …

- Incorporating 3D information
- Modeling complex interactions between actors and between actors and the scene
- Cross-modal seamless integration between text, audio, and visual information
- More attention to privacy and deep fakes detection
- …


humans
SCALE
talkens.ai
TALKENS.AI