

Chapter 1

Intermediate Blocks, AZee Templates and Pose Correction

Creativity in human expression often emerges from the interplay between structured rules and natural, fluid motion. In the context of (??) animation, while the AZee model offers a structured linguistic framework, it remains abstract and disconnected from the rich, dynamic nature of human motion. Natural human movement, including transitions and subtle nuances, is not dictated solely by linguistic constraints. It requires more than following a set of postures; it demands a synthesis that reflects the flexibility and fluidity inherent in real-life communication. To capture the full expressive power of ??, we must extend beyond AZee's formal descriptions and incorporate real-world data that can enhance motion and gesture authenticity.

In this chapter, we address this gap by focusing on two key areas: the generation of intermediate blocks using motion templates and the application of pose correction techniques. Intermediate blocks provide the smooth transitions between postures that are essential for natural movement, ensuring that the resulting animation feels fluid and coherent. Motion templates serve as pre-defined motion patterns that can guide these transitions, drawing from both artistic input and real-world data. Lastly, we introduce a pose correction module that leverages a ?? motion capture dataset to ensure that synthesized poses are contextually appropriate and realistic. Together, these components aim to enhance the naturalness of ?? animation, filling in the missing details left by the linguistic model and creating a more seamless and expressive experience.

In this chapter, section ?? discusses the generation of intermediate blocks using motion templates, while section ?? explores the application of pose correction techniques to the AZee low-level synthesis system. Section ?? presents the results of these techniques, highlighting the improvements in naturalness and coherence achieved through motion templates and pose correction. Finally,

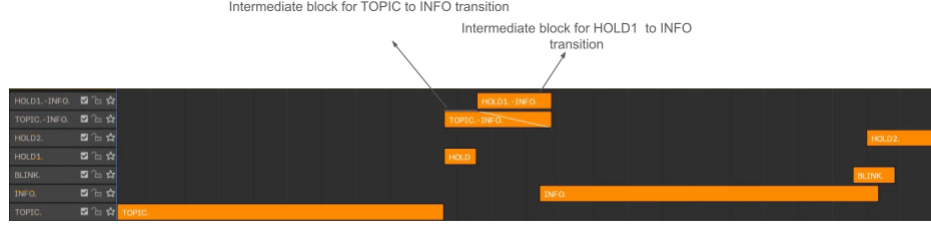


Figure 1.1: Intermediate Blocks for the AZee rule *info-about(topic,info)*

section ?? concludes the chapter and outlines future areas for improvement.

1.1 Intermediate Block Generation

Intermediate blocks are segments of motion not explicitly specified by the AZee model. Building on the concept of motion curves, as discussed earlier in Section ?? of Chapter ??, these blocks ensure smooth transitions between posture sequences. Figure ?? illustrates intermediate blocks generated for the AZee rule *info-about(topic,info)*. To create these blocks, we examine each motion segment B_1 in the set S , which includes constraint blocks, transpath blocks, hold blocks and pre-animated blocks. For each B_1 , we find the closest subsequent segment B_2 in S , ensuring that B_2 starts after B_1 ends. Among all possible B_2 candidates, the one with the earliest start time is selected, and a transition block is generated to connect B_1 and B_2 , filling in the motion gaps left by AZee's structure.

Simplest way to fill in intermediate blocks is to use linear or some other spline in these blocks. However, we propose the use of motion templates, which provide predefined motion patterns that can guide the interpolation within these blocks.

1.1.1 Identifying Motion Templates

Before filling in the intermediate blocks with some motion template, we need to identify them using an AZee template check (as discussed in section ?? of chapter ??). The template check is a top-down search algorithm that selects the best motion template to match on the AZee expression. Figure ?? shows the template check for the AZee expression *info-about(cat,cute)*. The algorithm starts by searching for the most specific template that matches the expression, then gradually broadens the search until a suitable template is found.

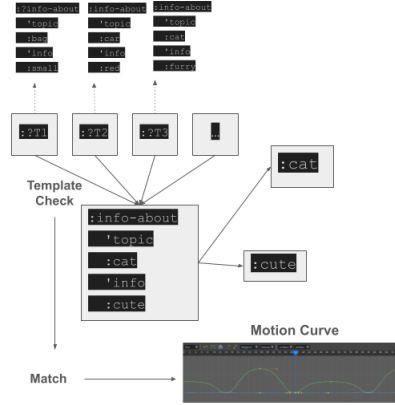


Figure 1.2: Top-Down Search for Motion Template

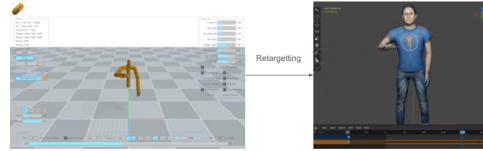


Figure 1.3: Retargeted mocap data to BAZeel avatar

1.1.2 Creating Motion Templates

Motion templates (as discussed in section ?? of chapter ??) are pre-defined motion patterns that guide the synthesis of new animations. In the context of AZee-driven synthesis, these templates serve as a blueprint for filling the intermediate blocks with relevant motion curves. For our experiments, we will be mainly using the *40 brèves* ? corpus. We chose to base this study on the first 5 rules in this corpus based on their frequency (see section ?? of annex).

Motion templates can be data-driven or artistic. Data driven templates are based on motion data extracted from the AZee rule and corresponding motion data. Artistically created templates, on the other hand, are designed by hand based on the AZee rule and desired motion features.

Data-Driven Motion Templates For creating data-driven templates, we first extract motion curves from the AZee annotated motion capture data. We use the Rosetta dataset ? which is already AZee annotated. We choose to only include the upper body bones in the dataset since AZee doesn't contain lower body bones. Next, we retargeted the ?? data on our BAZeel avatar, which is compatible with the AZee skeleton structure (figure ??).

We converted the ?? data into ?? pose arrays. The ?? pose array consists of

```
[[[ 9.99413192e-01 -3.03431898e-02 -1.57641545e-02 2.02674419e-03]
 [ 9.95326281e-01 -8.89065787e-02 -2.30883993e-02 2.98028998e-02]
 [ 1.00000000e+00 -7.45057971e-09 0.00000000e+00 -1.86264493e-09]
 ...
 [ 8.38357500e-01 1.69118538e-01 4.83241856e-01 -1.87170774e-01]
 [ 9.89919126e-01 4.00324985e-02 -1.35415837e-01 -1.09594055e-02]
 [ 9.99774754e-01 -5.62164048e-03 -1.32497288e-02 -1.55979460e-02]]

[[[ 9.99413192e-01 -3.03431898e-02 -1.57641545e-02 2.02674419e-03]
 [ 9.95326281e-01 -8.89065787e-02 -2.30883993e-02 2.98028998e-02]
 [ 1.00000000e+00 -7.45057971e-09 0.00000000e+00 -1.86264493e-09]
 ...
 [ 8.38357500e-01 1.69118538e-01 4.83241856e-01 -1.87170774e-01]
 [ 9.89919126e-01 4.00324985e-02 -1.35415837e-01 -1.09594055e-02]
 [ 9.99774754e-01 -5.62164048e-03 -1.32497288e-02 -1.55979460e-02]]

[[[ 9.99415074e-01 -3.02502699e-02 -1.57919303e-02 1.92501780e-03]
 [ 9.95448589e-01 -8.74409378e-02 -2.32355744e-02 2.99416557e-02]
 [ 1.00000000e+00 -7.48242002e-09 8.95502509e-12 -1.85070492e-09]
 ...
 [ 8.40322912e-01 1.71512499e-01 4.79522228e-01 -1.85739875e-01]
 [ 9.98134537e-01 3.77608687e-02 -1.34483531e-01 -1.10410759e-02]
 [ 9.99775052e-01 -5.97685575e-03 -1.32331504e-02 -1.54662952e-02]]]
```

Figure 1.4: AZee FK Pose Array

the rotation values of each joint in the AZee skeleton (figure ??). This representation is more suitable for our purposes since it allows us to directly create the templates based on the joint rotations.

To create the templates we used the affected bone chains based on the low level description of the rule. For example, for the rule *side-info*, we used the head, neck, and bust bones. We then extracted the motion curves for each bone and normalized them to the range $[-1, 1]$. The final blended rotations can be calculated using ?? between the n quaternions q_1, q_2, \dots, q_n :

$$q_{\text{blend}} = \text{Slerp}(\dots \text{Slerp}(\text{Slerp}(q_1, q_2, t_1), q_3, t_2) \dots, q_n, t_{n-1})$$

Where:

- t_1, t_2, \dots, t_{n-1} are the blend factors for each interpolation step, with $t_i \in [0, 1]$.
- The Slerp between two quaternions q_i and q_j is defined as:

$$\text{Slerp}(q_i, q_j, t) = \frac{\sin((1-t)\theta)}{\sin(\theta)} q_i + \frac{\sin(t\theta)}{\sin(\theta)} q_j$$

where $\theta = \arccos(q_i \cdot q_j)$ is the angle between the quaternions.

Figure ?? shows template creation process.

Artistically Created Motion Templates Motion templates can also be created artistically, based on the linguistic context and the desired expressive qualities. Artistically created template for the rule *about-ref* can be seen in figure ??.

For reference, figure ?? shows how the head and the spine chain move towards the reference point (??), while the neck compensates for the rotation. The motion template reflects the same.

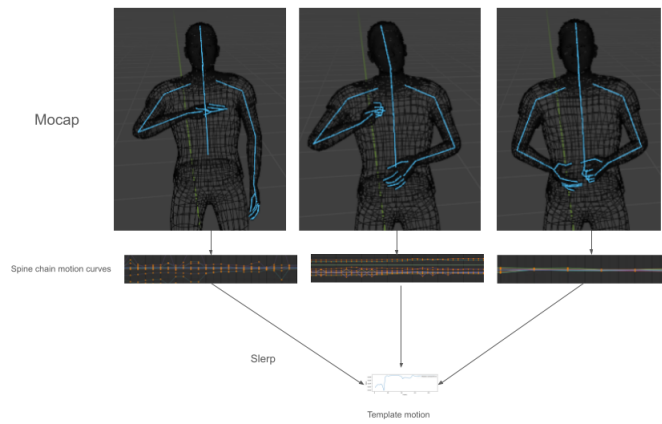


Figure 1.5: Data-Driven template creation for template *side-info*

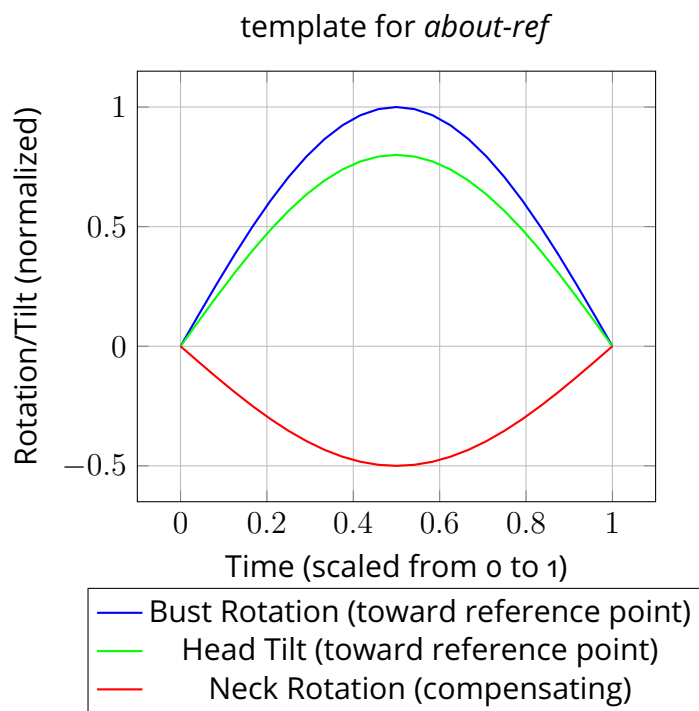


Figure 1.6: Artistically Created motion curves for template *about-ref*



S

Figure 1.7: Pose for the AZee rule *about-ref(:irak, Rssp)*

1.2 Pose Correction with AZee Low-Level synthesis

One part in enhancing the naturalness of ?? animations is to also ensure that the poses generated by the AZee system are contextually appropriate and realistic. Pose correction techniques can help achieve this by aligning the synthesized poses with the motion data in the dataset. In this section, we discuss the application of pose correction techniques to the AZee low-level synthesis system. By integrating pose correction, data-driven ??, and latent space representations into the AZee framework, we aim to enhance the realism and expressiveness of ?? animations.

1.2.1 Preparing the dataset

The first step in integrating pose correction into AZee is to train a ?? on a set of ?? poses. For this task, we use a dataset of ?? data collected from the Rosetta dataset ?. The dataset consists of 167066 poses which we assume to be representative enough to capture the diversity of poses and movements associated with different signs, providing a rich source of training data for the ??. For training, we use the same ?? pose array representation as discussed in section ??. We divided the dataset into training, validation, and test sets.

1.2.2 Training the Autoencoder

A ?? (earlier discussed in section ?? of chapter ??) is an autoencoder that learns a low-dimensional latent space representation of the input data. In the context of character animation, a ?? (like VPoser ?) can be used to capture the distribution of poses in a dataset, allowing for the generation of new poses that are statistically similar to the training data. The ?? consists of an encoder network that maps input poses to a latent space and a decoder network that reconstructs the input poses from the latent space (figure ??).

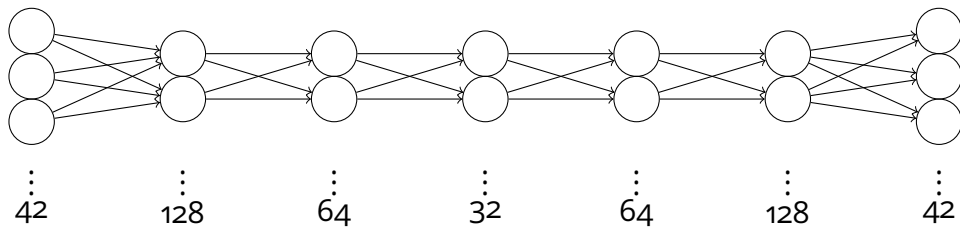


Figure 1.8: Architecture of the VAE

During training, batches of poses are processed, with each batch consisting of 512 samples. The training is carried out on a CUDA-enabled GPU, utilizing

mixed precision training for efficiency. The ?? architecture consists of a latent space with 32 dimensions, and the neural network has 512 neurons per layer.

The loss function used during training includes two primary components:

- **Reconstruction Loss:** This loss is computed at the joint level by comparing the reconstructed pose with the original pose using L1 loss. An additional pose-level reconstruction loss (L2 loss) is applied during the first 10 epochs to help the model learn better early on.
- **KL Divergence Loss:** The KL divergence regularizes the latent space by enforcing it to follow a standard normal distribution, ensuring that the latent space representation is smooth and continuous.

The total loss is the sum of the reconstruction loss and KL divergence loss. The optimization is carried out using the Adam optimizer with a learning rate of 1×10^{-2} and weight decay of 0.0001. A learning rate scheduler is used, which reduces the learning rate by a factor of 0.5 every third of the training epochs.

1.2.3 Implementing Pose Correction

With the ?? trained, we can now implement a pose correction system that leverages the learned latent space to match poses generated by the AZee system to the most appropriate pose in the dataset.

Algorithm ?? outlines the pose correction process. Given a target pose generated by the AZee synthesizer, we first encode the pose into the latent space using the ?? encoder. We then compute the distance between the encoded pose and each pose in the dataset, selecting the pose with the smallest distance as the best match. Finally, we decode the matched pose back into the AZee ?? pose array and apply it to the character.

1.3 Results and Implementation

1.3.1 Results and Implementation for Intermediate Block Generation

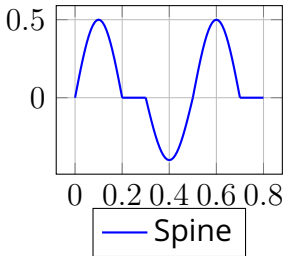
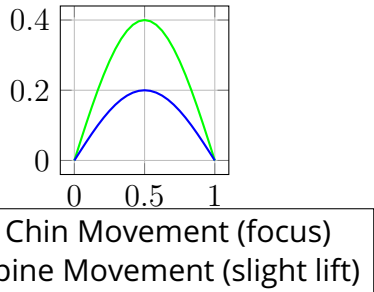
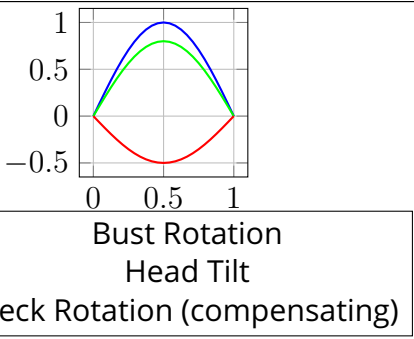
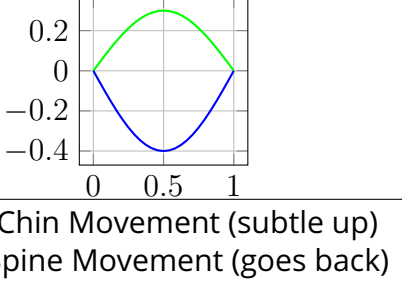
We chose to extend our previous animator in blender to include the intermediate block generation process. The intermediate blocks are represented as strips in the ??.

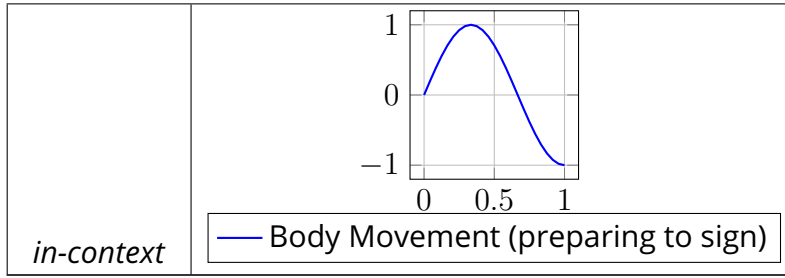
Table ?? shows the results of the intermediate block generation process. The table shows the motion curve profiles for the most common AZee rules.

Algorithm 1 AZee constraint optimization with pose correction algorithm

```
1: for frame in frames do
2:   switch_cursor_to_frame(f)
3:   for parallel_block in self.parallel_blocks do
4:     constraints.add(parallel_block.constraints)
5:   end for
6:   for constraint in constraints do
7:     constraint.apply(frame)
8:   end for
9:   model.pose_embedding
10:  model.global_trans
11:  optimizer = ...
12:  for epoch in range(max_epochs) do
13:    optimizer.zero_grad()
14:    ...
15:    optimizer.step()
16:    if loss.item() < threshold then
17:      break
18:    end if
19:  end for
20:  posture.keyframe(frame)
21: end for
```

Table 1.1: Motion curves of some intermediate blocks for some AZee rules

AZee Rule	Template
<i>info-about</i>	 <p>— Spine</p>
<i>side-info</i>	 <p>— Chin Movement (focus) — Spine Movement (slight lift)</p>
<i>about-ref</i>	 <p>— Bust Rotation — Head Tilt — Neck Rotation (compensating)</p>
<i>instance-of</i>	 <p>— Chin Movement (subtle up) — Spine Movement (goes back)</p>



We compare the results of using standard interpolation techniques with the proposed template-based interpolation method in the video ¹.

An increase in naturalness when animating using the intermediate blocks technique can be seen. Moreover, this work provides us insights regarding the motion profile which an AZee rule abstracts.

1.3.2 Results and Implementation for Pose Correction

We created the pose corrector as a separate module in the existing AZee animator. The pose corrector is responsible for matching the poses generated by the AZee system to the most appropriate pose in the dataset. As the algorithm ?? shows, the pose correction process is applied just after the AZee constraints are optimized.

Snapshots with standard synthesis and synthesis with pose correction and the corresponding AZee code for the same are shown in table ??.

The pose correction system seems to produce more natural and contextually appropriate animations compared to standard joint-limit based synthesis. However, due to retargeting losses, the integration of pose correction into ?? synthesis is still in the early stages. We also observe that the corrector might change the pose of the character in a way that is not always desirable ??.

Lastly, figure ?? shows how retargeting the ?? data to the AZee skeleton structure results in a loss of information. This loss can affect the quality of the generated animations and is an area for future improvement.

1.4 Conclusion and Future Work

In this chapter, we explored techniques to make our synthesis system more natural. By leveraging the AZee model and existing motion data, we were able to fill

¹https://github.com/Paritosh97/phd/raw/master/supplementary_material/ch5_templates.mp4

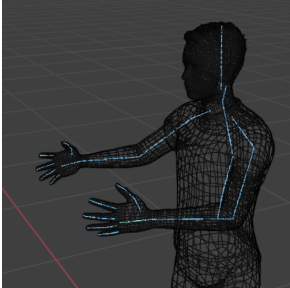
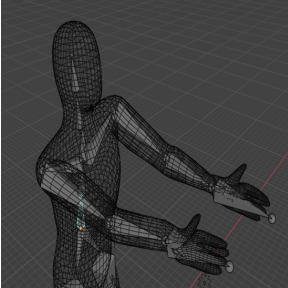
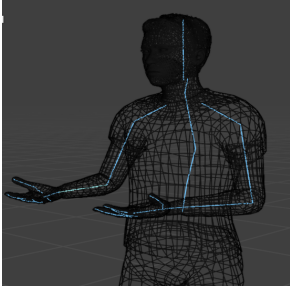
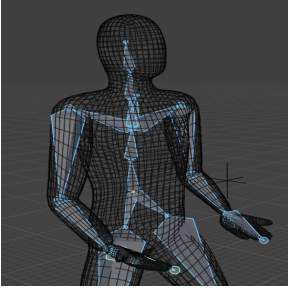
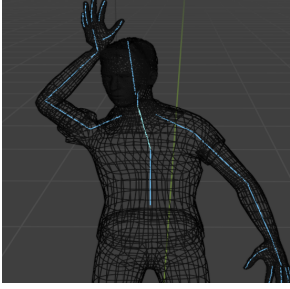
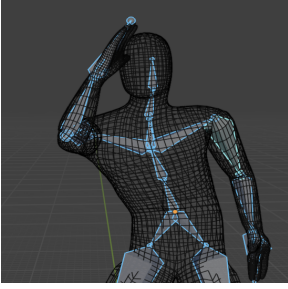
Standard Synthesis	Synthesis with Pose Correction	AZee Code
		<i>:armoire</i>
		<i>:maintenant</i>
		<i>:about-ref(:irak, Rssp)</i>

Table 1.2: Comparison of standard synthesis and synthesis with pose correction

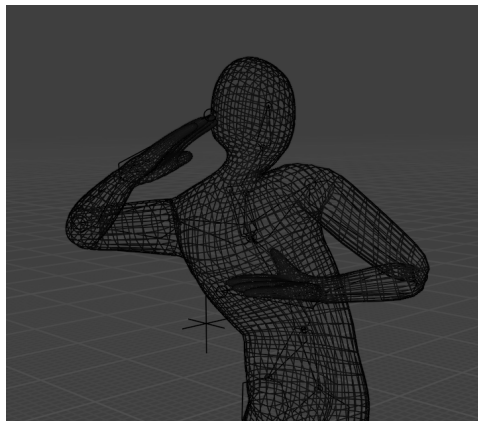


Figure 1.9: Problems with current correction module

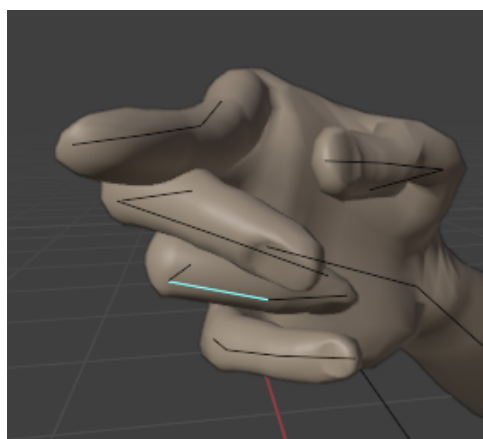


Figure 1.10: Retargeting losses for pointing on the Rssp

in the gaps left by the linguistic model and create more expressive and coherent animations.

While the current system of motion templates proves effective for generating smooth transitions in most body movements, it faces significant challenges when dealing with finer details such as finger articulation. The complexity of finger movements in ??, especially during rapid transitions, requires high precision in motion curves and templates. Unfortunately, the current implementation lacks the necessary granularity, resulting in robotic transitions. Moreover, capturing the nuances of coarticulation between handshapes be it with ?? or using artistic templates an open challenge. Another limitation stems from too much data in the motion templates. This might bring in information such as the identity of the signer, which may not always generalize well. This can be mitigated by using a more generalized set of motion templates that can be applied across different signing scenarios.

Similarly, while the integration of pose correction into ?? synthesis in our animator, it also introduces new challenges. Data-driven and latent space methods typically require significant computational resources, both during training and inference. This can be a major barrier in applications where low latency is critical. Also the effectiveness of deep learning models depends heavily on the availability of high-quality training data. In many cases, obtaining sufficient ?? data can be difficult, especially for non-standard or stylized animations. Lastly, while deep learning models can generate realistic and high-quality animations, they often lack the fine-grained control that human animators require. Ensuring that these models produce outputs that align with artistic vision remains a significant challenge.

Future areas to improve our work technique could include a more detailed

quantitative analysis of motion capture data based on a broader range of AZee rules, as well as further refinement of finger tracking and facial expressions in the motion templates. Since obtaining good quality ?? data is also a challenge, pose estimation based on image data could be explored as an alternative ??.

Since, motion curves for intermediate blocks might not have one-to-one correspondence with the AZee rules, newer techniques such as Diffusion based inbetweening ? could also be used for better intermediate block generation.

Similarly, newer posers based on neural distance fields ? or diffusion ? could be used for pose correction since the current poser has a bayesian bias. Also, continuity of the trained model with respect to signing spaces could be studied further improving the learnt pose prior. Lastly, since natural human gestures are independant of the language, a more generalized pose correction model could be trained on a more generalized dataset.