WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

REAL TIME IMAGE ANIMATION

Praneeta T. R, Preethi S, Sarathram M, Dr. J. B. Jona, Mr. S. A. Gunasekaran

**[a]Msc. Decision and Computing Sciences, Department of Computing, Coimbatore Institute of Technology, Coimbatore , India*
**[b]Associate Professor, Department Of Computer Applications, Coimbatore Institute of Technology, Coimbatore, India*
*. *[c] Assistant Professor, Department Of Computer Applications, Coimbatore Institute of Technology, Coimbatore, India.*

A B S T R A C T

Image animation consists of generating a video sequence so that an object in a source image is animated according to the motion of a driving video. Our framework addresses this problem without using any annotation or prior information about the specific object to animate. Once trained on a set of videos depicting objects of the same category (e.g. faces, human bodies), our method can be applied to any object of this class. To achieve this, we decouple appearance and motion information using a self-supervised formulation. To support complex motions, we use a representation consisting of a set of learned key points along with their local affine transformations. A generator network models occlusions arising during target motions and combines the appearance extracted from the source image and the motion derived from the driving video. Our framework scores best on diverse benchmarks and on a variety of object categories.

Keywords*: Image Animation, CNN, Transformations, Deep fake.*

1. INTRODUCTION

Creating videos by animating objects in still images has several applications such as remote control movie production, photography, and e-commerce. In other words, image animation is nothing but the task of automatically generating videos by combining the appearance extracted from a source image with motion patterns derived from a driving video. For instance, a face image of a certain person can be animated following the facial expressions of another individual. Most of the methods tackle this problem by considering strong priors on the object representation and resorting to computer graphics techniques These approaches can be referred to as object-specific methods, as they consider the knowledge about the model of the specific object to animate.

Off late, deep generative models are the most efficient techniques for image animation and video generation using Computer Vision and Neural Networks for Real Time Data. However, these methods are dependent on pre-trained models so as to extract the keypoint locations.

2. APPROACH

Deepfake technology is generally used to create fake content, replace faces,voice, and handle emotions. Besides this, we can also digitally imitate an action by a person who never performed. For training purposes, we fetched a large number of videos consisting of the objects of the same object category. Our model is trained to reconstruct the training videos by combining a single frame and a learned latent representation of the motion in the video. Observing frame pairs (source and driving), each extracted from the same video, it learns to encode motion as a combination of motion-specific keypoint displacements and local affine transformations. At test time we apply our model to pairs composed of the source image and of each frame of the driving video and perform image animation of the source object.

An overview of our approach is presented in Figure3 below. Our framework is composed of two main modules: the motion estimation module and the image generation module. The purpose of the motion estimation module is to predict a dense motion field. We assume there exists an abstract reference frame. And we independently estimate two transformations: from reference to source and from reference to driving. This choice allows us to independently process source and driving frames. This is desired since, at test time the model receives pairs of the source image and driving frames sampled from a different video, which can be very different visually.

In the first step, we approximate both transformations from sets of sparse trajectories, obtained by using key points learned in a self-supervised way. We model motion in the neighborhood of each keypoint using local affine transformations. Compared to using keypoint displacements only, the local affine transformations allow us to model a larger family of transformations. During the second step, a dense motion network combines the local approximations to obtain the resulting dense motion field. Furthermore, in addition to the dense motion field, this network outputs an occlusion mask that indicates which image parts of driving can be reconstructed by warping of the source image and which parts should be painted (inferred from the context). Finally, the generation module renders an image of the source object moving as provided in the driving video. Here, we use a generator

network that warps the source image according to dense motion and in paints the image parts that are occluded in the source image. Once trained, it applies to arbitrary objects of the same category. For faces It successfully extracts and retargets Facial expressions like head poses and eye movement.

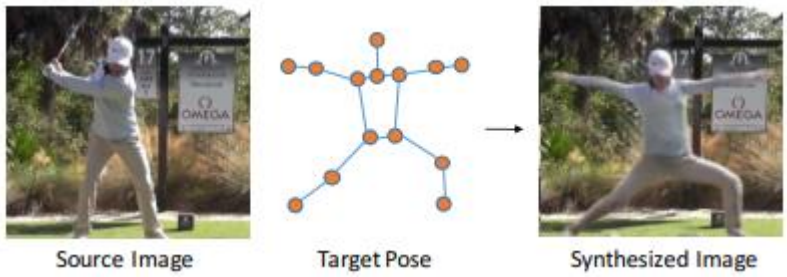

Fig 1. Represents the retaining of the apperrence

Figure 1. Our method takes an input image along with a desired target pose, and automatically synthesizes a new image depicting the person in that pose. We retain the person's appearance as well as filling in appropriate background textures.

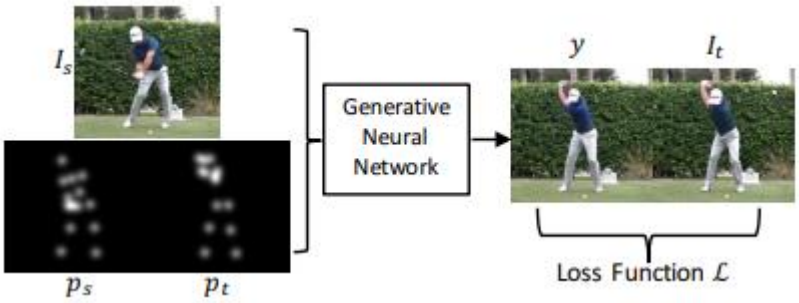

Fig 2. Synthesizes an image

Figure 2. This network takes as input a tuple of the form (Is, ps, pt), and synthesizes an image y. During training, a loss function L is used to minimize error between y and It. We visualize ps and pt here as single-channel images, though in our model they contain a separate channel for each joint

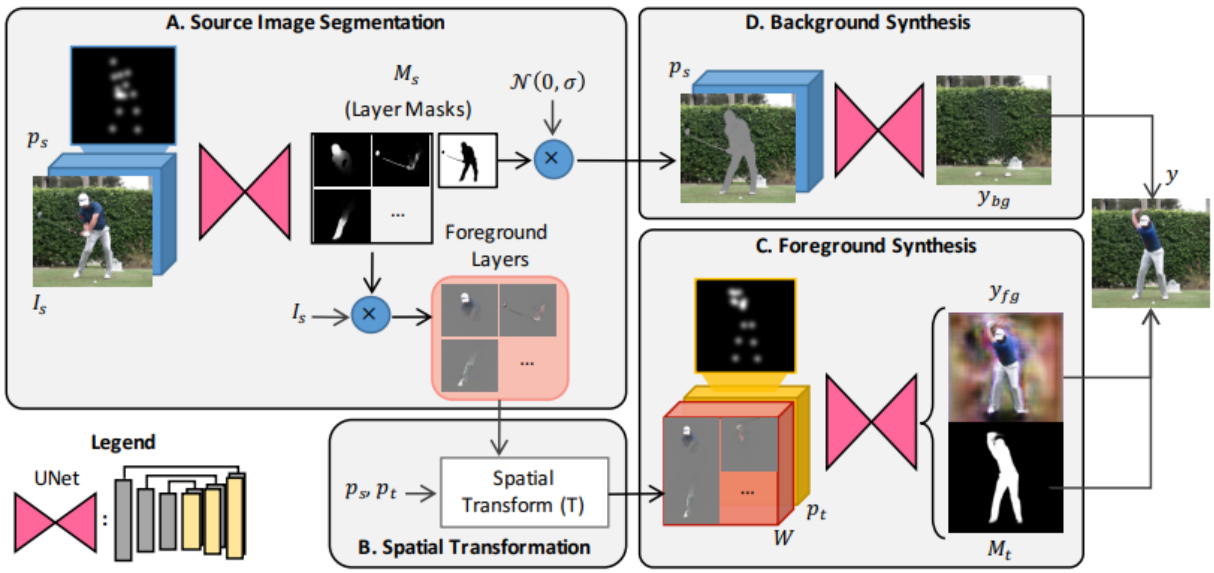

Fig 3. Network Architecture

Figure 3. Network architecture, consisting of four modules. Module A performs image segmentation on input, separating the person's body and objects held by the person from the background. Module B spatially transforms the body parts in obtained input. Module C synthesizes a target foreground image by fusing the body parts in a realistic manner. This module also simultaneously outputs a foreground mask Mt. Module D synthesizes a background image, via hole-filling. Finally, we composite outputs from C & D to produce y.

Fig 4. Sample Model Output

Figure 4 depicts the Sample output produced by the model for corresponding inputs.
The processes said earlier are achieved by the below mentioned terms with help of Neural Network such as Convolution Neural networks.

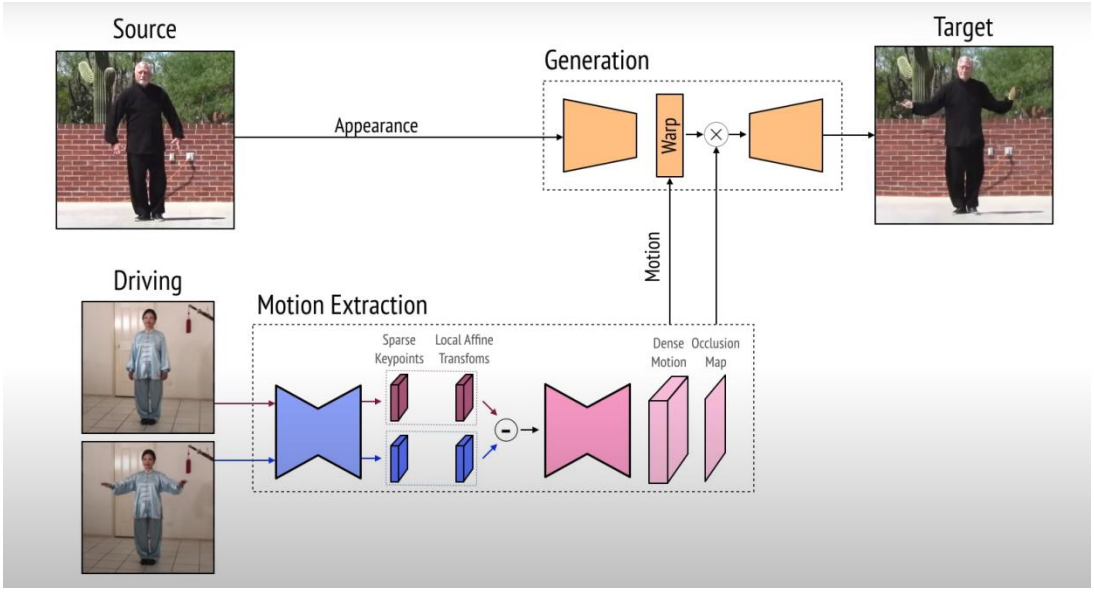

Fig. 5. Represents work model for Train Data

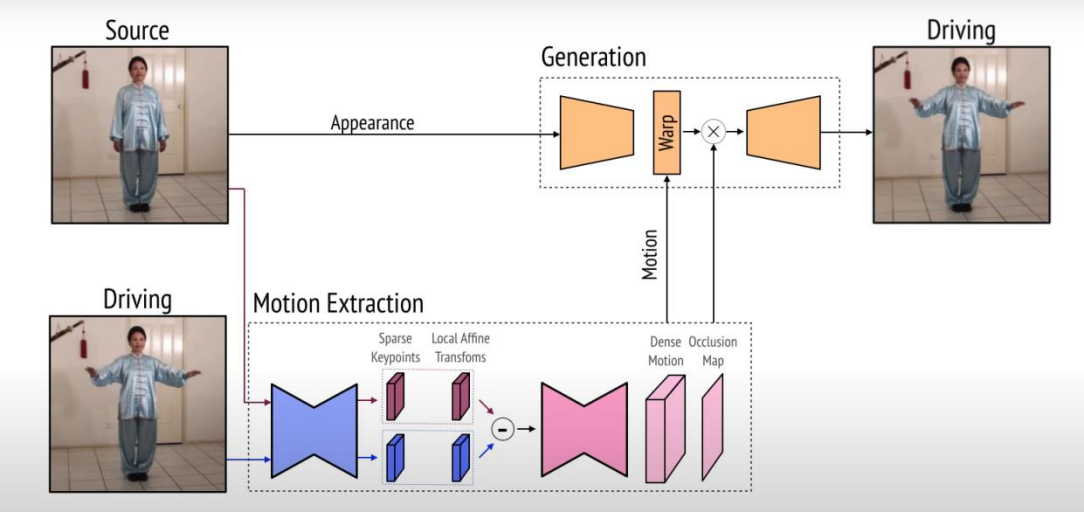

Fig. 6. Represents work model for Test Data

3. DATASET

Our dataset for handling this project were taken from kaggle and google images which contains some pictures and videos for source and driving inputs. We train and test our method on theser different datasets containing various objects. Nearly forty photos and six videos are used for handling the project.

Fig. 7. Sample of our Data Set

4. PROCESS

4.1. Skin Color Segmentation And Tracking

The color of human skin is a striking feature to track and to robustly segment the operator's hands and face. It is exploited that human skin color is independent of the human race and on the wavelength of the exposed light. The same observation can be made considering the transformed color in common video formats. Hence, the human skin-color can be defined as a “global skin-color cloud” in the color space. This is utilized successfully in a fast and robust region-growing based segmentation algorithm. The skin color segmentation is performed on predefined thresholds in the U,V-space of the video signal. Then, a blob recognition identifies the hands and the head in the sub-sampled image. Based on this information, a region growing approach segments the complete skin-color region of the hands and the head quite accurately.

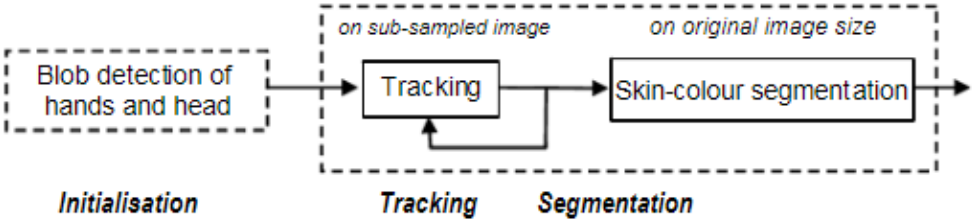

Fig. 8. Block diagram of the segmentation and tracking method of the hands.

4.2. Facial Feature Extraction

The aim of facial feature tracking is to obtain sufficient information in order to derive a convincing and reliable rotation of the operator's head. As a result from the segmentation and tracking algorithm described in the previous section, the bounding box of the operator's head is used as a starting point for facial feature extraction. The skin-coloured pixels inside the bounding box are marked and a standard feature tracker is applied to this limited face region. The feature tracker is based on a two- step approach. First, relevant features are selected by using corner operators such as Moravec or Harris detectors. Secondly, the selected features are then tracked continuously from frame to frame by using a feature dissimilarity measure.

This guarantees that features are discarded from further tracking in the case of occlusions. Even in the case of a rotating head some good features become distorted due to perspective changes or even become invisible and get lost. In Fig. 3, markers of selected features are shown in the face region in three succeeding frames. The big cross assigns the median value of all skin coloured pixels. The considered skin color region is marked by the line around the face. Due to the blond hairs of the test person, the hairs are recognized as well as skin.

Fig. 9. Facial feature tracking result of three succeeding frames

4.3. Texture Fitting

A well-known method in computer graphics, texture mapping improves virtual objects' quality by applying real images onto them. Its low cost in terms of computation time proves very useful for real-time applications. For virtual humans, the texture can add grain to the skin, including details like variations in the hair and mouth color. These features require correlation between the image and the 3D object. A simple projection doesn't always realize this correlation: the object designed by hand can differ slightly from the real image. An interactive fitting of the texture is required. The program enables the designer to interactively select a few 3D points on the object, which are then projected onto the 2D image. The projection can be chosen and set interactively, hence the designer can adjust these projected points to their correct position on the image. This method obtains the texture coordinates for the selected 3D points.

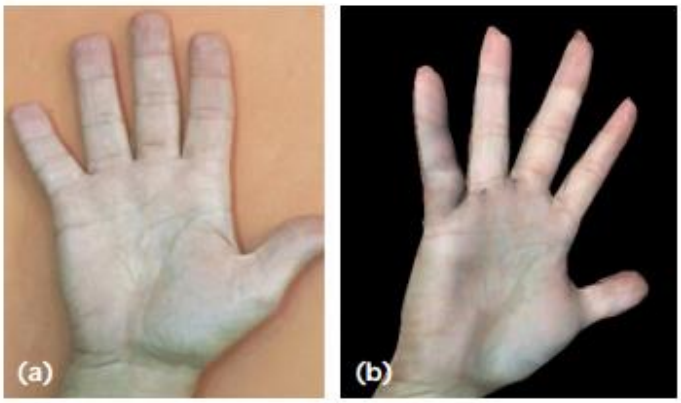

Fig. 10. (a) texture is applied to (b).

4.4. Animating the Body

A real-time virtual human is one that can act at the same speed as a real person. Virtual reality, interactive television, and games require real-time virtual-human bodies. The generally accepted approach models the body in two layers, skeleton and skin; a third layer— cloth—could also be added. The skeleton layer consists of a tree-structured, fixed-topology hierarchy of joints connecting limbs, each with minimum and maximum limits. The skin layer, attached to the skeleton, generates the skin surfaces of the body. Animation of the virtual body affects the skeleton layer. Animation of skin and cloth are automatically computed by deforming or transforming vertices. This means that the skeleton animation doesn't normally depend on the two other layers and could be defined in very different ways.

Fig. 11. Represents the way, Body is animation and re-structured

4.5. Body Deformations

Few attempts at producing virtual humans have reached the right compromise between realism and animation speed. On the one hand, some applications emphasize speed and interaction. Typically, they use a polygonal representation: the skin wrapped around the skeleton is represented with a fixed mesh divided at important joints where deformations occur. Because no deformations are computed within a body part—that is, between two joints—the virtual human appears "rigid" and lacks realism. Moreover, visually distracting artifacts may arise at joints where two body parts connect, for example when limbs are bent.

On the other hand, some applications stress visual accuracy. Such applications generally compute the skin from implicit primitives and use a physical model to deform the body's envelope. Though this approach yields very satisfactory results in terms of realism, it proves so computationally demanding that it's unsuitable for real-time applications. We investigated a third approach that combines some elements of the previous ones, allowing a good tradeoff between realism and rendering speed. Our simple yet powerful system smoothly deforms the skin, greatly enhancing the human appearance of our virtual characters while preserving a high frame rate to meet the real-time requirements of virtual environments.

5. METHODOLOGY

For training, we employ a large collection of video sequences containing objects of the same object category. Our model is trained to reconstruct the training videos by combining a single frame and a learned latent representation of the motion in the video. Observing frame pairs, each extracted from the same video, it learns to encode motion as a combination of motion-specific keypoint displacements and local affine transformations. At test time we apply our model to pairs composed of the source image and of each frame of the driving video and perform image animation of the source object.

5.1. Local Affine Transformations

The motion estimation module estimates the backward optical flow Ts←D from a driving frame D to the source frame S. As discussed above, we propose to approximate Ts←D by its first order Taylor expansion in a neighborhood of the keypoint locations. In the rest of this section, we describe the motivation behind this choice, and detail the proposed approximation of Ts←D. We assume there exist an abstract reference frame R. Therefore, estimating Ts←D consists in estimating Ts←R and Ts←D. Furthermore, given a frame X, we estimate each transformation Ts←R in the neighborhood of the learned keypoints.

In practice, Ts←R(pk) and Td←R(pk) in are predicted by the keypoint predictor. More precisely, we employ the standard U-Net architecture that estimates K heatmaps, one for each keypoint. The last layer of the decoder uses softmax activations in order to predict heat maps that can be interpreted as keypoint detection confidence maps. Each expected keypoint location is estimated using the average operation. For both frames S and D, the keypoint predictor network also outputs four additional channels for each keypoint. From these channels, we obtain the coefficients of the matrices by computing spatial weighted average using as weights the corresponding keypoint confidence map.

5.2. Combining Local Motions

Employ a convolutional network P to estimate Tˆs←D from the set of Taylor approximations of Ts←D(z) in the key points and the original source frame S. Importantly, since Tˆs←D maps each pixel location in D with its corresponding location in S, the local patterns in Tˆs←D, such as edges or texture, are pixel-to-pixel aligned with D but not with S. This misalignment issue makes the task harder for the network to predict Tˆ S←D from S. In order to provide inputs already roughly aligned with Tˆs←D, we warp the source frame S according to local transformations estimated. Thus, we obtain K transformed images S 1 , . . . S K that are each aligned with Tˆ S←D in the neighborhood of a keypoint. Importantly, we also consider an additional image S 0 = S for the background. For each keypoint pk we additionally compute heatmaps Hk indicating the dense motion network where each transformation happens. Each Hk(z) is implemented as the difference of two heatmaps centered in Td←R(pk) and Ts←R(pk).

5.3. Occlusion Image Generation

The source image S is not pixel-to-pixel aligned with the image to be generated Dˆ . In order to handle this misalignment, we use a feature warping strategy. More precisely, after two down-sampling convolutional blocks, we obtain a feature map $\xi \in R^{H0 \times W0}$ of dimension H0 × W0 . We then warp ξ according to Tˆs←D. In the presence of occlusions in S, optical flow may not be sufficient to generate Dˆ . Indeed, the occluded parts in S cannot be recovered by image-warping and thus should be inpainted. Consequently, we introduce an occlusion map to mask out the feature map regions that should be in-painted. Thus, the occlusion mask diminishes the impact of the features corresponding to the occluded parts. We estimate the occlusion mask from our sparse keypoint representation, by adding a channel to the final layer of the dense motion network.

5.4. Imposing Equivariance Constraints

There is no need for any keypoint in our prediction and also while training as well which may result in an unstable performance. Equivariance constraint is one of the most important factors driving the discovery of unsupervised keypoints [18, 43]. It forces the model to predict consistent keypoints with respect to known geometric transformations. We use thin plate splines deformations as they were previously used in unsupervised keypoint detection [18, 43] and are similar to natural image deformations. We assume that an image X undergoes a known spatial deformation Tx←Y. In this case Tx←Y can be an affine transformation or a thin plane spline deformation. After this deformation we obtain a new image Y. Now by applying our extended motion estimator to both images, we obtain a set of local approximations for Tx←R and Ty←R.

6. OUTPUTS

These images show that the image has been animated, and a video has been generated for input source video which is merged with help of driving video.

7. CONCLUSION

This approach for image animation is based on keypoints and local affine transformations. This Formulation describes the motion field between two frames and is efficiently computed and in this way, motion is described as a set of keypoints displacements and local affine transformations. A generator network combines the appearance of the source image and the motion representation of the driving video. In addition, we proposed to explicitly model occlusions in order to indicate to the generator network which image parts should be in-painted. We evaluated the proposed method both quantitatively and qualitatively and showed that our approach clearly outperforms state of the art on all the benchmarks.

Further research includes elaborating on a user-interface for real-time simulation and improving the simulated individuals visual quality. Increasing realism requires revising and improving our methods, although the results should not differ much qualitatively. We're working on the real-time simulation of hair and deformable clothing, and on a variety of autonomous behaviors. With the goal of accelerating the cloning process, we're also making progress on the automatic 3D reconstruction and simulation of virtual faces.

REFERENCES

[1] Guha Balakrishnan, Amy Zhao, Adrian V Dalca, Fredo Durand, and John Guttag. Synthesizing images of humans in unseen poses. In CVPR, 2018.

[2] Z. Liu, R. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In Proc. CVPR, 2017.

[3] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In NIPS, 2016.

[4] https://www.researchgate.net/publication/221356279_Real-Time_Avatar_Animation_Steered_by_Live_Body_Motion.

[5] Dinghuang Ji, Max McFarland, Silvio Savarese. Deep view morphing. Computer Vision and Pattern Recognition (CVPR), 2017.

[6] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 2650–2658, 2015.

[7] R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, 2003.

[8] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. Image and Vision Computing, 2010.