Motion Synthesis by Motion Decomposition

Yash Shah, Arjun Jain
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay
{yashshah, ajain}@cse.iitb.ac.in

Abstract

Encoder-decoder based models have been widely used for predicting future frames of 3D human motion given a prior context. However, such models are not able to generate motion unconditionally, as well as they struggle to predict frames far in the future. Generative models are apt for this task, but most of the recent work on them treats motion as an irreducible sequence of joint vectors. In this report, we propose a novel method of decomposing motion sequences into a series of bands, with each one serving as a 'texture' over the previous one, and also introduce new theoretical latent variable models based on this decomposition. We provide experimental results on one of these models that uses neural ODEs, and leave others for future research.

1. Introduction

Motion prediction is becoming increasing important for the domains of human-machine interaction, crowd tracking and monitoring, obstacle avoidance and many more. Since no physical law can govern the conscious movements of a person, there is a dire need of a statistical model that can approximate this behavior. This presents a tremendous challenge for machine learning algorithms.

Several discriminative models have already been proposed for this task. However, they have difficulty in predicting actions far in time, and the sequence often converges to a static pose. Moreover, these models ignore stochasticity of human motion arising due to the biological nature of human body. Due to these reasons, a generative model is more appropriate for modelling its behavior.

However, all the generative models so far treat human motion as an irreducible sequence of joint vectors. This treatment overlooks the fact that for a given task, the associated motion sequence is mostly similar except for small subject-specific variations. For example, all humans walk in almost the same manner. Separating the base and varying components from a motion sequence can definitely help to model it better.

2. Related Work

All prior work done in the field of motion prediction can be broadly classified into encoder-decoder (ERD) based models and generative models. While the former directly predicts sequences based on the given context, the latter models the distribution of the input, from which sequences are then sampled. ERD models are deterministic in the sense that given the same input sequence, they will predict the same output sequence. Generative models, on the other hand, are much more flexible but difficult to train.

The ERD model was proposed by Katerina et al. [4] for recognition and prediction of human body pose in videos and motion capture. Their model incorporated nonlinear encoder and decoder networks before and after recurrent layers and was able to jointly learn representations and their dynamics. Martinez et al. [6] improved upon existing architecture by proposing residual variants, removing teacher-forcing and allowing multiple actions to be modelled.

FAIR's QuaterNet [7] was another step forward in this domain. They proposed models for both short- and long-term predictions by representing rotations with quaternions and penalizing absolute position errors instead of angle errors. They also addressed the issues of error accumulation along the kinematic chain and discontinuities arising due to parameterizations.

In the generative model domain, stochastic recurrent networks and VAE-LSTMs have been popular choices. Among these, the work by Habibie et al. [5] has been a major inspiration. They proposed a novel model in the variational inference framework that allowed users to produce animations from high-level control signals; this solved the ambiguity problem for long sequences, and even reduced the predictive error drastically.

The inspiration behind motion decomposition is the work of Pullen et al. [8]. They used motion capture data to enhance animation by adding detail to various degrees of freedom, in a process they called texturing. They used Laplacian pyramids to convert sequences of joint angles into frequency bands, and used correlation amongst joint angles to fill in the missing degrees of freedom.

3. Motion Decomposition

3.1. General formulation

Let $X = \{x_t\}_{t=1}^{T}$ be a sequence of joint vectors $x_t \in \mathbb{R}^d$ representing 3D human motion, and let f be a smoothing filter. We obtain a series of smoothed sequences $\{U_i\}_{i=1}^{k}$ by performing repeated 1D temporal convolutions as follows

$$\begin{aligned} U_1 &= X \\ U_i &= f \circledast U_{i-1},\ 2 \le i \le k \end{aligned} \tag{1}$$

We then obtain k bands representing decomposed motion sequences of X as follows

$$\begin{aligned} X^s &= U_k \\ X^i &= U_{k-i} - U_{k-i+1},\ 1 \le i \le k-1 \end{aligned} \tag{2}$$

It is easy to observe that $X = X^s + \sum_{i=1}^{k-1} X^i$. X^s being the smoothest serves as the base sequence, and $\{X^i\}_{i=1}^{k-1}$ as textures that are progressively added to X^s to obtain the observed sequence X. This separation of the base and noise components should intuitively allow us to model motion better, with the former capturing underlying task-specific sequence, and the latter capturing subject-specific variations.

For simplicity, we model these bands using independent Gaussian random variables. Thus, for time t we have

$$\begin{aligned} x_t^s &\sim \mathcal{N}(\mu_{s,t}, \sigma_{s,t}^2) \\ x_t^i &\sim \mathcal{N}(\mu_{i,t}, \sigma_{i,t}^2),\ 1 \le i \le k-1 \end{aligned} \tag{3}$$

which gives

$$x_t \sim \mathcal{N}(\mu_t, \sigma_t^2) \tag{4}$$

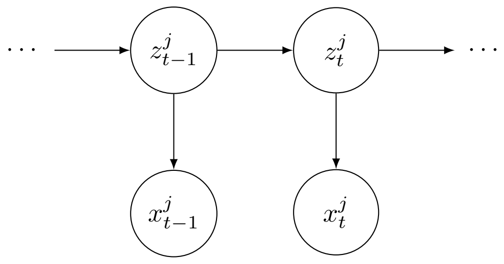
where $\mu_t = \mu_{s,t} + \sum_{i=1}^{k-1} \mu_{i,t}$ and $\sigma_t^2 = \sigma_{s,t}^2 + \sum_{i=1}^{k-1} \sigma_{i,t}^2$. This result is obtained using a property of independent Gaussian random variables and the fact that $x_t = x_t^s + \sum_{i=1}^{k-1} x_t^i$.

3.2. Introducing latent variables

In order to express this idea using the variational framework, we introduce latent variables z_t^j so that the sequence $\{x_t^j\}_{t=1}^{T}$ is generated from them according to the dynamics shown in Figure 1 (here $j \in \{s, 1 \dots k-1\}$). The manner in which the distributions $q_\phi(z|x)$, $p(z)$ and $p_\theta(x|z)$ are computed depends on the model used.

4. Neural ODEs - A Failed Attempt

Neural Ordinary Differential Equations (NODEs) were introduced as a new family of deep neural networks and provided a radically different approach for learning to model continuous-time sequential data. Unlike standard recurrent neural networks, which require discretizing observation intervals, continuously-defined dynamics that are used in NODEs can naturally incorporate data which arrives at arbitrary times. Since the same property allowed NODEs to generate arbitrarily long sequences, we were convinced that using NODE blocks in our latent variable model could result in a much more powerful generative model than the current state-of-the-art.



$$\begin{tikzcd} \cdots \arrow[r] & z_{t-1}^j \arrow[r] \arrow[d] & z_t^j \arrow[r] \arrow[d] & \cdots \\ & x_{t-1}^j & x_t^j & \end{tikzcd}$$

Figure 1: Dynamics of sequence generation.

4.1. Neural Ordinary Differential Equations

NODEs parameterize the derivative of the hidden state using a neural network. The output of NODE is computed by a black-box differential equation solver, whose operations can be backpropagated through, by evaluating the hidden unit dynamics f wherever necessary to determine the solution with the desired accuracy. Mathematically, given the initial state $\mathbf{h}(0)$, NODE learns the function f specified as

$$\frac{d\mathbf{h}(t)}{dt} = f(\mathbf{h}(t), t, \theta) \tag{5}$$

Starting from the initial state $\mathbf{h}(0)$, the NODE can get $\mathbf{h}(t)$ at any time t as the solution to this initial value problem at time t.

NODE provides a continuous-time, generative approach for modeling time series by representing them using a latent trajectory. Each trajectory is determined from a local initial state, $\mathbf{z}_{t_0}$, and a shared global set of latent dynamics f, parameterized using a neural network. Given observation times $t_0, t_1, \dots, t_N$ and an initial state $\mathbf{z}_{t_0}$, an ODE solver produces $\mathbf{z}_{t_1}, \dots, \mathbf{z}_{t_N}$, which describe the latent state at each observation. Mathematically,

$$\mathbf{z}_{t_0} \sim p(\mathbf{z}_{t_0}) \tag{6}$$

$$\mathbf{z}_{t_1}, \dots, \mathbf{z}_{t_N} = \text{ODESolve}(\mathbf{z}_{t_0}, f, \theta_f, t_0, t_1, \dots, t_N) \tag{7}$$

$$\mathbf{x}_{t_i} \sim p(\mathbf{x} \mid \mathbf{z}_{t_i}, \theta_x) \tag{8}$$

To obtain the latent representation $\mathbf{z}_{t_0}$, the sequence is traversed in reverse using an RNN and parameters of the distribution $q(\mathbf{z}_{t_0} | \{\mathbf{x}_{t_i}, t_i\}_i, \theta_{enc})$ are obtained. Then, a standard VAE algorithm with an RNN variational posterior and an ODESolve model is followed:

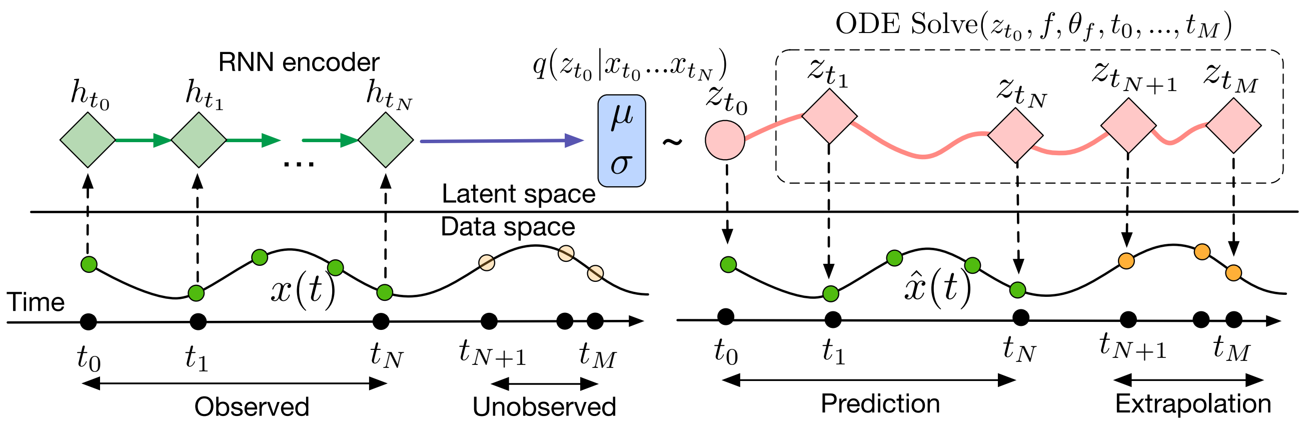


Figure 2: Computation graph of the latent ODE model. Borrowed from [2].

1. Run an RNN encoder through the time series and infer the parameters for a posterior over $\mathbf{z}_{t_0}$:

$$q(\mathbf{z}_{t_0}|\{\mathbf{x}_{t_i}, t_i\}_i, \phi) = \mathcal{N}(\mathbf{z}_{t_0}|\mu_{\mathbf{z}_{t_0}}, \sigma_{\mathbf{z}_0})$$

where $\mu_{\mathbf{z}_0}, \sigma_{\mathbf{z}_0}$ come from the hidden state of RNN$(\{\mathbf{x}_{t_i}, t_i\}_i, \phi)$

2. Sample $\mathbf{z}_{t_0} \sim q(\mathbf{z}_{t_0}|\{\mathbf{x}_{t_i}, t_i\}_i)$

3. Obtain $\mathbf{z}_{t_1}, \mathbf{z}_{t_2}, \ldots, \mathbf{z}_{t_M}$ by solving ODESolve$(\mathbf{z}_{t_0}, f, \theta_f, t_0, \ldots, t_M))$, where f is the function defining the gradient $\frac{d\mathbf{z}}{dt}$ as a function of $\mathbf{z}$.

4. Maximize ELBO given by

$$\sum_{i=1}^{M} \log p(\mathbf{x}_{t_i}|\mathbf{z}_{t_i}, \theta_x) + \log p(\mathbf{z}_{t_0}) - \log q(\mathbf{z}_{t_0}|\{\mathbf{x}_{t_i}, t_i\}_i, \phi)$$

where $p(\mathbf{z}_{t_0}) = \mathcal{N}(\mathbf{0}, \mathbf{1})$

This method is summarized in Figure 2.

4.2. Model

We built our latent variable model borrowing ideas from section 4.1 and extending them to support the motion decomposition approach discussed in section 3.

We use separate encoder RNNs and latent ODE blocks for each band obtained on decomposing the motion sequence. This follows from the assumption that the latent variables $\{z_t^s\} \cup_i \{z_t^i\}$ are independent of each other, and that $\{x_t^s\} \cup_i \{x_t^i\}$ are conditionally independent of each other given $\{z_t^s\} \cup_i \{z_t^i\}$.

The modified algorithm is as follows:

1. Run the k RNN encoders through each band obtained by decomposing the motion sequence and infer the parameters for posteriors over $\mathbf{z}_{t_0}^j$ where $j \in \{s, 1 \ldots k-1\}$:

$$q(\mathbf{z}_{t_0}^j|\{\mathbf{x}_{t_i}^j, t_i\}_i, \phi_j) = \mathcal{N}(\mathbf{z}_{t_0}^j|\mu_{\mathbf{z}_{t_0}}^j, \sigma_{\mathbf{z}_0}^j)$$

where $\mu_{\mathbf{z}_0}^j, \sigma_{\mathbf{z}_0}^j$ come from the hidden state of RNN$(\{\mathbf{x}_{t_i}^j, t_i\}_i, \phi_j)$.

2. Sample $\mathbf{z}_{t_0}^j \sim q(\mathbf{z}_{t_0}^j|\{\mathbf{x}_{t_i}^j, t_i\}_i)$ for all j and concatenate the sampled values to get $\mathbf{z}_{t_0} = ||_j\ \mathbf{z}_{t_0}^j$.

3. Obtain $\mathbf{z}_{t_1}, \mathbf{z}_{t_2}, \ldots, \mathbf{z}_{t_M}$ by solving ODESolve$(\mathbf{z}_{t_0}, f, \theta_f, t_0, \ldots, t_M))$, where f is the function defining the gradient $\frac{d\mathbf{z}}{dt}$ as a function of $\mathbf{z}$. Note that f is such that

$$f(\mathbf{z}) = ||_j f^*(\mathbf{z}^j)$$

That is, it computes derivatives of $\mathbf{z}$ separately and then concatenates them; this ensures that $\mathbf{z}^j$'s remain independent of each other.

4. Maximize ELBO given by

$$\sum_j \sum_{i=1}^{M} \Bigg(\log p(\mathbf{x}_{t_i}^j|\mathbf{z}_{t_i}^j, \theta_{x^j}) + \log p(\mathbf{z}_{t_0}^j) - \log q(\mathbf{z}_{t_0}^j|\{\mathbf{x}_{t_i}^j, t_i\}_i, \phi_j) \Bigg)$$

where $p(\mathbf{z}_{t_0}^j) = \mathcal{N}(\mathbf{0}, \mathbf{1})$.

4.3. Training

We trained and evaluated our model on Human3.6M dataset. Sequences were divided into chunks of length 512, and then grouped into batches of size 64. We used $k = 3$

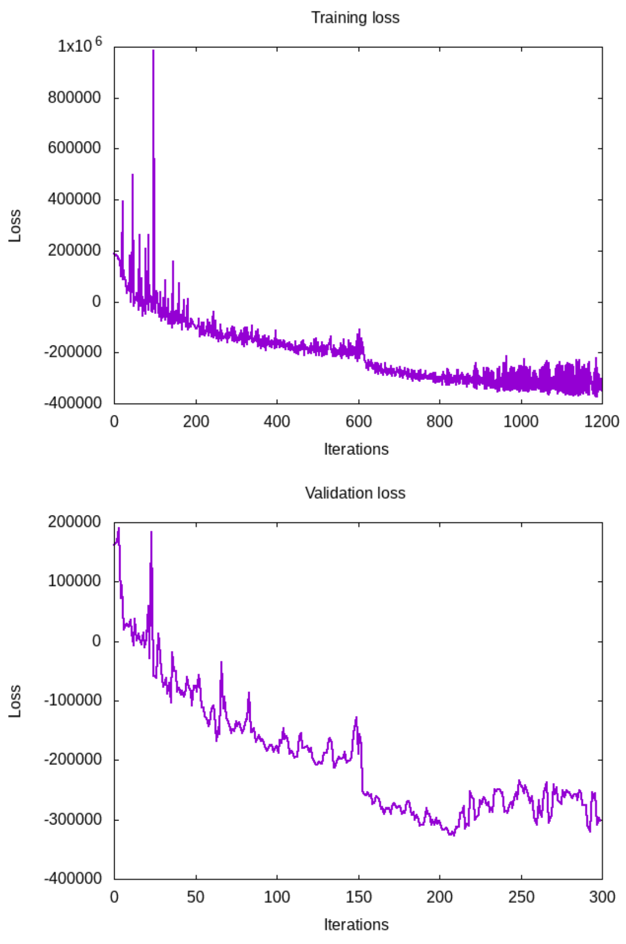
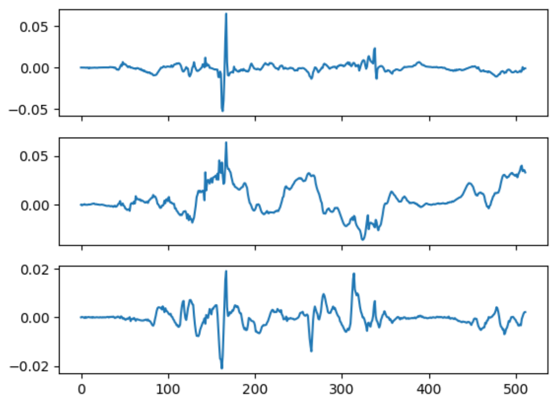
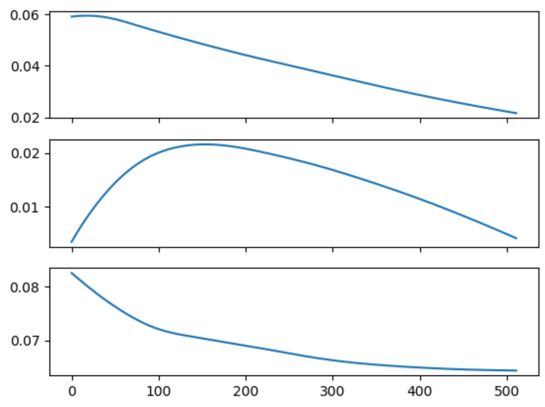


Figure 3: Variation in loss with iterations, $k = 3$



(a) Ground truth sequences



(b) Maximum likelihood generated sequences

Figure 4: Model after 100 epochs of training, $k = 3$

in our experiments; that is, one base band and two texture bands. The input to our model were 96-dimensional joint vectors consisting of 3 euler angles per joint. The latent hidden dynamics f was modeled using a feedforward neural network with hidden dimension 128 and latent dimension 96. SGD with initial learning rate 0.01 and momentum 0.9 was used for optimizing the ELBO described in section 4.1. We followed a learning-rate schedule that decayed it by half every 50 epochs. We have released our codebase[1] for future research.

4.4. Experimental results

Using NODE out-of-the-box after only slight modifications did not help at all. The loss, which is defined to be the negative of ELBO term, decreased progressively (see Figure 3) without any improvement in the quality of samples generated. Even after 100 epochs of training, the maximum likelihood sequences generated were drastically different than the ground truth as can be seen in Figure 4.

We attributed this failure to the inherent nature of NODE and the manner in which it had been used for this task. Since NODE learns to approximate the derivative of hidden state, it is expected to produce 'smooth' outputs. Clearly, the sequences that we were trying to model were non-differentiable at many points, as can be seen from Figure 4a. Thus, using NODE as it is didn't prove to be useful.

5. Latent Variable Models

Much prior work has been done to build generative models for sequences based on the variational framework. We now examine two such models theoretically, and provide their variants that are coupled along with the motion decomposition idea.

[1]https://github.com/ys1998/motion-forecast

5.1. Variational RNN

The variational recurrent neural network, or VRNN, was proposed by [3] as a recurrent latent variable model for sequential data. We provide below the modified steps, using the same notation as before, for handling with decomposed motion sequences.

1. First decompose the motion sequence into k bands as described in section 3. For each band, the following steps should be performed.
2. Compute parameters of the prior and approximate posterior distributions using the hidden state and input at time t as shown

$$[\mu_{j,t},\ \sigma_{j,t}] = \phi^{prior}(\mathbf{h}^j_{t-1})$$
$$[\mu_{z^j,t},\ \sigma_{z^j,t}] = \phi^{enc}(\phi^{\mathbf{x}^j}(\mathbf{x}^j_t), \mathbf{h}^j_{t-1})$$

3. Sample $\mathbf{z}^j_t$ from the approximate posterior distribution

$$\mathbf{z}^j_t \sim \mathcal{N}(\mu_{z^j,t},\ \text{diag}(\sigma^2_{z^j,t}))$$

4. Using the sampled $\mathbf{z}^j_t$, compute the parameters of the generating distribution $p(\mathbf{x}^j_t | \mathbf{z}^j_{\leq t}, \mathbf{x}^j_{<t})$ as shown

$$[\mu_{x^j,t},\ \sigma_{x^j,t}] = \phi^{dec}(\phi^{\mathbf{z}^j}(\mathbf{z}^j_t), \mathbf{h}^j_{t-1})$$

5. Compute the hidden state $\mathbf{h}^j_t$ as shown

$$\mathbf{h}^j_t = f_\theta(\phi^{\mathbf{x}^j}(\mathbf{x}^j_t), \phi^{\mathbf{z}^j}(\mathbf{z}^j_t), \mathbf{h}^j_{t-1})$$

6. Maximize the ELBO given below using the parameters of the respective distributions computed above

$$\sum_j \sum_{i=1}^{T} \Bigg(\log p(\mathbf{x}^j_t \mid \mathbf{z}^j_{\leq t}, \mathbf{x}^j_{<t}) - \\ \text{KL}(q(\mathbf{z}^j_t \mid \mathbf{x}^j_{\leq t}, \mathbf{z}^j_{<t}) \mid\mid p(\mathbf{z}^j_t \mid \mathbf{x}^j_{<t}, \mathbf{z}^j_{<t}) \Bigg)$$

5.2. Stochastic Recurrent Networks

Stochastic Recurrent Networks, or STORN, were proposed by [1]. They are similar to VRNNs, except the fact that they restrict the latent variables to a fixed prior distribution of $\mathcal{N}(\mathbf{0}, \mathbf{1})$. They have already been tried on motion capture data for imputing missing values, and have proved quite effective. Their extension to decomposed motion sequences is similar to that for VRNNs — use separate STORNs for each band, but maximize the combined ELBO term. We refer the reader to the section on VRNNs and [1] for more details.

6. Conclusion

In this report, we introduced a novel method of modelling motion sequences by decomposing them into base and texture bands. We test the approach on the recently proposed latent ODE model, and highlight the reasons of our failure on the same. We also propose extensions to existing generative models based on the variational framework which incorporate the decomposition step into themselves, and also new ELBO terms for the same.

Acknowledgements

I am grateful to my guide, Prof. Arjun Jain, for his constant support throughout this project especially during times when we were not getting expected results. I am also grateful to Rishabh Dabral and Rahul Mitra for clearing several of our doubts, providing relevant reading material and also providing us access to computational resources for our experiments.

References

[1] J. Bayer and C. Osendorfer. Learning stochastic recurrent networks. *CoRR*, abs/1411.7610, 2014. 5

[2] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 2018. 3

[3] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. A recurrent latent variable model for sequential data. In *NIPS*, 2015. 5

[4] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik. Recurrent network models for human dynamics. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4346–4354, 2015. 1

[5] I. Habibie, D. Holden, J. Schwarz, J. Yearsley, and T. Komura. A recurrent variational autoencoder for human motion synthesis. In *BMVC*, 2017. 1

[6] J. Martinez, M. J. Black, and J. Romero. On human motion prediction using recurrent neural networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4674–4683, 2017. 1

[7] D. Pavllo, D. Grangier, and M. Auli. Quaternet: A quaternion-based recurrent model for human motion. In *BMVC*, 2018. 1

[8] K. Pullen and C. Bregler. Motion capture assisted animation: texturing and synthesis. In *SIGGRAPH*, 2002. 1