Data-Driven Reconstruction of Human Locomotion Using a Single Smartphone

Haegwang Eom, Byungkuk Choi and Junyong Noh

Graduate School of Culture Technology, KAIST, Republic of Korea

Abstract

Generating a visually appealing human motion sequence using low-dimensional control signals is a major line of study in the motion research area in computer graphics. We propose a novel approach that allows us to reconstruct full body human locomotion using a single inertial sensing device, a smartphone. Smartphones are among the most widely used devices and incorporate inertial sensors such as an accelerometer and a gyroscope. To find a mapping between a full body pose and smartphone sensor data, we perform low dimensional embedding of full body motion capture data, based on a Gaussian Process Latent Variable Model. Our system ensures temporal coherence between the reconstructed poses by using a state decomposition model for automatic phase segmentation. Finally, application of the proposed nonlinear regression algorithm finds a proper mapping between the latent space and the sensor data. Our framework effectively reconstructs plausible 3D locomotion sequences. We compare the generated animation to ground truth data obtained using a commercial motion capture system.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Three-Dimensional Graphics and Realism]: Animation—

1. Introduction

Motion capture systems have been widely used in computer animation fields to create natural human motion easily and accurately. However, commercial motion capture systems require special equipment that is often too costly for general usage. Requirements such as attaching retro-reflective markers to a body or relying on infrared cameras further impose spatial constraints. In this paper, we present a novel method for reconstructing a user's motion using a single smartphone. As our method is robust against noise and does not require accurate sensor data, we simply utilize a smartphone, one of the most widely used, low-cost, and portable devices as a sensing device. To express human poses using only one smartphone, we create a statistical model from pre-recorded human motion data. Once a motion model with high quality data is available, new input smartphone sensor data can be recorded anywhere, and by anyone, without any spatial constraints.

Our approach combines a couple of machine learning algorithms to address the challenge of mapping directly between all joint channels and sensor data. Our low-dimensional motion model controls the human pose in response to new sensor data. Low dimensional embedding is performed through the Gaussian Process Latent Variable Model (GPLVM). Use of the GPLVM generates a new latent space in which each pose is effectively spread over, based on the correlation between the poses. Latent coordinates defined by this nonlinear probabilistic model make the mapping robust against noise.

Reconstructing motions pose by pose may cause ambiguity and temporal incoherence. To avoid these problems, we employ a Hidden Markov Model (HMM) for phase segmentation of new input data from a smartphone. Based on the trained HMM classifiers, the phase at each frame is identified automatically by utilizing a moving window. For each phase, we apply a Multilayer Perceptron (MLP) model to define a mapping between the latent points and the training sensor data that belong to the phase.

The main contributions of this research can be summarized as follows. From a system point of view, our approach utilizes a single smartphone for the purpose of capturing full body motion. The reconstruction can be performed without spatial constraints or special equipment while ensuring reasonable quality. From a technical point of view, because the

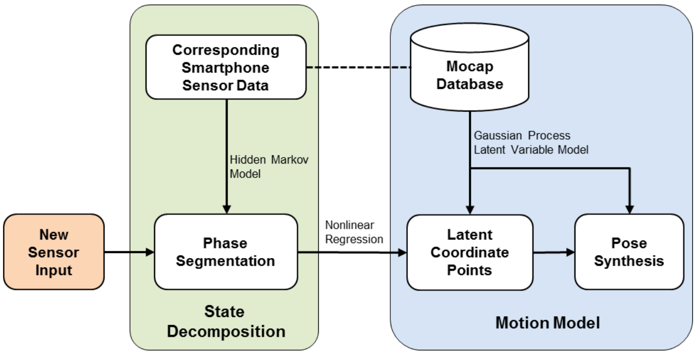


Figure 1: System overview

statistical data model produces reasonable interpolation results for missing data, our approach does not rely on a large dataset. In addition, our mapping algorithm effectively relates pre-recorded motion clips with sensor data that have a limited degree of freedom, based on low-dimensional embedding. An equally important contribution is establishing a state decomposition model, which facilitates learning on a semantic level. Based on this, we successfully handle pose ambiguity such as shaking artifacts by utilizing phase classification.

2. Related Work

Approaches that focus on extracting meaningful motion data using a small number of sensors have been developed. In early research, Freeman et al. [FTOK96] described applications of such approaches specifically for the game field. The arcade game Police 911 [Kon01] utilized infrared sensors for tracking and rendering hand motion. EyeToy [Son03] employed a vision-based system to track simple arm motion in 2D space without relying on explicit markers. Slyper and Hodgins [SH08b] attached several accelerometers on an actor's body to express upper body motion using their search algorithm.

Many studies have focused on reconstructing full body motion from low dimensional sensor inputs. Badler et al. [BHG93] applied real time inverse kinematics to control a standing character using the data captured by four magnetic sensors. Semwal et al. [SHS98] employed eight magnetic sensors along with an analytic mapping function. Yin and Pai [YP03] utilized a foot pressure sensor to develop an interface to generate pose motion. Chai and Hodgins [CH05] presented an online control system that employed a couple of synchronized cameras and several markers. Their online local model utilized various pre-recorded captured data. Pons-Moll et al. [PMBH*10] presented a hybrid tracker that combined a video with a small number of inertial sensors.

Tautges et al. [TKZW] experimented the tracking of human body motions while changing the number of inertial sensors, using the extended method proposed by Cai and Hodings [CH05]. Later, Tautges et al. [TZK*11] suggested a method for high-quality full body motion reconstruction using four accelerometers, each of which was attached to the end of each limb. They experimented for different cases while changing the numbers and positions of the sensors, and decided four accelerometers as the minimum number that produces various smooth motions. Unlike our method, their approach using only one accelerometer produces temporal incoherence and unsteadiness which are apparently shown in their video.

Generating motion in real time sometimes requires a reduction of degree of freedom. Oore et al. [OTH02] employed a pair of tracking devices that have six degrees of freedom to control stepping and walking activities. Layered acting for character animation was suggested in Dontcheva et al. [DYP03]. Shiratori and Hodgins [SH08a] analyzed parameters for physically based character motion utilizing a commercial gaming controller. To overcome limitations caused by the use of linear regression, Lawrence [Law04] proposed a nonlinear dimension reduction approach based on a GPLVM. This technique has been applied to human motion data and extended to continuous character control [GMHP04] [WFH08] [LWH*12] [UFG*08].

Our approach is similar to previous methods in that the goal is to generate high-quality full body locomotion. However, unlike Chai and Hodgins [CH05], we do not require markers or cameras and eliminate spatial constraints during the capture step. Our approach does not rely on search and playback of data from a database, as Slyper and Hodgins [SH08b] did. Different from Tautges et al. [TZK*11], we focus on reconstructing human locomotion. Although we only utilize a single smartphone, the results are smooth and plausible. In addition, our method shows robustness to speed variation.

3. Overview

Figure 1 shows how our system transforms a small number of sensor signals from a single smartphone to full body motion. We train a probabilistic model using both pre-recorded high quality 3D motion and synchronized smartphone sensor data. We prepared motion capture data and matching sensor data captured simultaneously for various activities such as straight walking and jumping. Capture was performed using a Vicon system [Sys02] with eight 120Hz Mx-40 cameras. The captured motion of each activity consists of only a few hundred frames and they are used for learning.

In the training step, we utilize a HMM for phase detection and a GPLVM to represent full body motion in a low dimensional space. An MLP algorithm is then utilized as nonlinear regression to map both spaces. To handle new input data, our state decomposition system first determines a phase at every frame. Next, MLP identifies new latent points and the GPLVM synthesizes new poses including root translation. Finally, common animation transformation operations including inverse kinematics and foot-skate clean-up are used to automatically modify the synthesized animation to smoothly satisfy additional constraints such as foot placement.

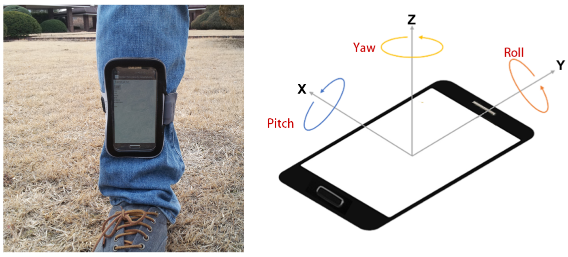


Figure 2: The left image shows the placement of a smartphone, and the right image describes the local coordinates information.

4. Sensor Recording

We use a Samsung Galaxy S4 smartphone for sensor data recording, although any smartphone equipped with an accelerometer and a gyroscope would be equally adequate for the present purpose. As we focus on the reconstruction of locomotion, we attach the smartphone near the ankle using an armband. We try to infer the associated upper body motion from the data of the leg for locomotion.

Calibrated data are specified with the unit of $\mathrm{m/s^2}$ for the accelerometer and rad/s for the gyroscope. All sensor readings are recorded with respect to the local coordinate system of the sensor. Figure 2 shows the placement of the smartphone and the local coordinate axes of the internal sensors. We assume a feature vector can represent the pose of an actor. It includes XYZ axes for the accelerometer and yaw, roll, and pitch axes for the gyroscope.

4.1. Post-Processing

Post-processing of the captured data from the smartphone sensors is necessary because the data contain abundant noise. Sensor data are not always reliable and the time unit of recording is not constant. To overcome this problem, we sample the data in a fixed time interval (e.g. 30fps) to obtain feature vectors and linearly interpolate the data for the remaining portions. The sampled data undergo noise removal before they are fed into the learning process along with synchronized motion capture sequences. Application of a noise reduction algorithm like a low-pass filter achieves this. Experimentally, we found out that applying a 1-D Gaussian filter to each dimension effectively cleans up the data. Figure 3 shows graphs before and after the noise reduction.

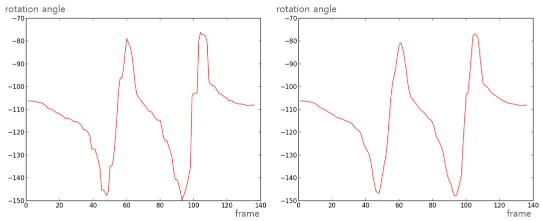


Figure 3: The left image represents raw input data from sensor, and the right image shows the filtering results.

5. Motion Model

In this section, we describe the learning between character motion and sensor data. Character motion generally requires a high dimensional description and it is difficult to match the motions directly with low dimensional data obtained from a smartphone. Therefore, we create a new low-dimensional space that represents high quality training motion examples. Use of low-dimensional embedding similar to Levine et al. [LWH*12] serves this purpose. A probabilistic model based on a GPLVM [Law05] represents pose data with latent coordinate points. A nonlinear regression method then associates these latent variables with the data from a smartphone.

5.1. Pose Likelihood Term

The pose vector y is composed of each channel value of joint angles including the root. In order to define a mapping from low-dimensional latent variables x to pose vectors y, we use a Gaussian Process (GP) model. After training the system with corresponding (x, y) data, the GP model tries to predict the likelihood of a new vector y for a new input vector x. To account for a different range of variance for each joint, the channels of the pose vector are scaled up by a diagonal matrix $\mathrm{W} = \mathrm{diag}\left(w_1, \ldots, w_{d_y}\right)$, where d_y is the dimension of y [GMHP04]. To consider correlation among data,

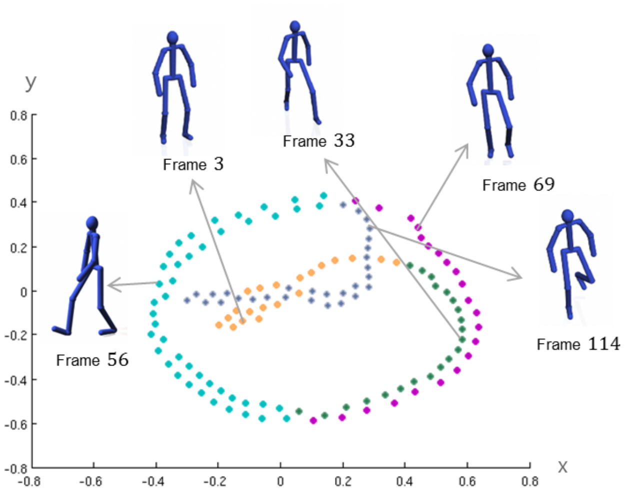


Figure 4: Two dimensional latent space of the GPLVM learned from a motion clip of two steps walking. Colors represent each phase; orange (from start until taking the left foot off the ground), green (from the left foot moving until taking the right foot off (in the starting part)), cyan (from the right foot moving until taking the left foot off (during walking)), purple (the opposite of cyan), gray (from taking the left foot off the ground to stop)

we choose a radial basis function (RBF) kernel with the parameter $\vec{\alpha}$:

$$k_{\mathrm{rbf}}\left(x_i, x_j; \vec{\alpha}\right) = \alpha_1 \exp\left(-\frac{\alpha_2}{2}\left|\left|x_i - x_j\right|\right|^2\right) + \alpha_3\delta_{ij} \quad (1)$$

where δ_{ij} is 1 when i and j are the same and 0 otherwise. With the RBF kernel, the correlation falls off smoothly when the distance between two latent points increases. We define a kernel matrix $K_{\mathcal{Y}}$ that consists of $K_{Y(i,j)} = k(x_i, x_j)$. The log likelihood term about the pose is represented by $\ln p\left(Y|X, W, \vec{\alpha}\right)$, which is proportional to the following equation.

$$L_Y = -\frac{1}{2}\mathrm{tr}\left(K_Y^{-1} Y W^2 Y^T\right) - \frac{d_y}{2}\ln|K_Y| + N\ln|W| \quad (2)$$

where N is the number of training data, $Y = [y_1, \ldots, y_N]^T$ and $X = [x_1, \ldots, x_N]^T$.

5.2. Velocity Gaussian Process

Considering a pose frame by frame may cause temporal incoherence. In addition, incorporating the root position is important to achieve a complete character motion. However, directly including the relative root translation to Y causes artifacts such as sliding effects or incorrect mapping results. To remedy this, we utilize the degree of movement per time step. We compute a new GP in the velocity domain using the velocity matrix $\dot{Y}$. Here, the root velocity in the form of horizontal world translation is also considered [LWH*12]. The log likelihood of the velocity GP requires a new kernel function:

$$k_{\dot{Y}}\left(\left[x_i, x_{i-1}\right], \left[x_j, x_{j-1}\right]; \vec{\beta}\right) = \beta_1 \dot{x}_i^T \dot{x}_j \exp\left(-\frac{\beta_2}{2}\left|\left|\dot{x}_i - \dot{x}_j\right|\right|^2 - \frac{\beta_3}{2}\left|\left|x_i - x_j\right|\right|^2\right) + \beta_4\delta_{ij} \quad (3)$$

where $\dot{x}_i = x_i - x_{i-1}$ and $\vec{\beta}$ denotes the parameters that will be optimized. Similar to Equation (2), the log likelihood value is proportional to the following equation:

$$L_{\dot{Y}} = -\frac{1}{2}\mathrm{tr}\left(K_{\dot{Y}}^{-1} \dot{Y} W_{\dot{Y}}{}^2 \dot{Y}^T\right) - \frac{d_{\dot{y}}}{2}\ln\left|K_{\dot{Y}}\right| + N\ln\left|W_{\dot{Y}}\right|, \quad (4)$$

$$L_{\dot{Y}} \propto \ln p\left(\dot{Y}|X, W_{\dot{Y}}, \vec{\beta}\right) \quad (5)$$

where $\dot{Y} = [\dot{y}_1, \ldots, \dot{y}_{N-1}]^T$ and $W_{\dot{Y}}$ is a scaling matrix. Incorporating the velocity GP model in the learning process produces better embedding results and leads to more reasonable character movements.

5.3. Model Learning

In the learning process, a setup is required for parameters such as latent variables X, hyper parameters for kernel functions $\vec{\alpha}$, $\vec{\beta}$, and scaling matrices $W, W_{\dot{Y}}$. Maximizing the log posterior that includes input pose data Y and its velocity $\dot{Y}$ achieves this.

$$\begin{gathered}\ln p\left(X, \vec{\alpha}, \vec{\beta}, W, W_{\dot{Y}}|Y, \dot{Y}\right) \propto \\ L_Y + L_{\dot{Y}} + \ln p\left(\vec{\alpha}\right) + \ln p(\vec{\beta})\end{gathered} \quad (6)$$

L_{Y} and $\mathrm{L}_{\dot{\mathrm{Y}}}$ are the pose and velocity likelihood terms, respectively, from Equations (2) and (4), and represent original motion information. $\ln p\left(\overrightarrow{\alpha}\right) = -\sum_i \ln\alpha_i$ and $\ln p(\overrightarrow{\beta}) = -\sum_i \ln\beta_i$ are prior terms for the parameters from the kernel functions. The log posterior is maximized by applying the LBFGS algorithm. The gradients of each likelihood and the priors for the parameters are also calculated in this step [WFH08].

5.4. Hidden Markov Model

Given sensor data S, where S is an $\mathrm{N} \times \mathrm{k}$ matrix with k sensor data dimensions, and its corresponding latent variable X from our motion model, a mapping between S and X using standard non-linear regression techniques produces a poor result as the data cannot be assumed to be independent and identically distributed. To accurately describe the sequential property of our data, the mapping function from S to X should be trained in semantic time pieces to ensure temporal coherence.

First, similar to Min et al. [MCC09], we manually divide training sequences into a few phases based on foot contact states. Figure 4 illustrates the state decomposition in the latent space from the GPLVM, where each color represents a different phase. The phase separation prevents possible ambiguity by imposing phase information in the pose reconstruction step.

To make our HMM training tractable, we first discretize sensor data S by using a small number of features instead of real values. Generally, if inputs to HMM are continous, a simple solution is to discretize the inputs and use the discretized inputs as training features [Alp04]. To construct a matrix C, which is a discretized version of S, we simply compute the slope of the sensor data for each dimension at every time step, and encode the slope as an integer value in a uniform interval, as described in Figure 5. As our discretization scheme considers the temporal tendency of each phase, even a small number of feature values (3 integers in our experiments) can sufficiently represent a distinctive feature vector and can be used for comparing a certain phase to others.

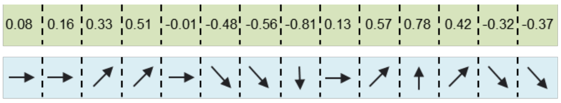


Figure 5: The first row shows the slope of an arbitrary 1 dimensional sensor data. The second row represents the discretized version of the first row. We use integer values instead of arrows in our implementation.

The HMM must be trained for each phase. For each training, C is rearranged to C' based on the manually identified class label of the phase. The resulting C' finally consists of m $\mathrm{c_i}$ block matrices, where m represents the number of phases and each row of $\mathrm{c_i}$ represents a discretized sensor data vector that is classified as the ith phase, as described in Figure 6. Note that each $\mathrm{c_i}$ block matrix contains the semantically same type of motion. By applying the HMM to each $\mathrm{c_i}$ block matrix, we obtain the total number of m HMM classifiers that will be used to evaluate new sensor data in Section 6.1.

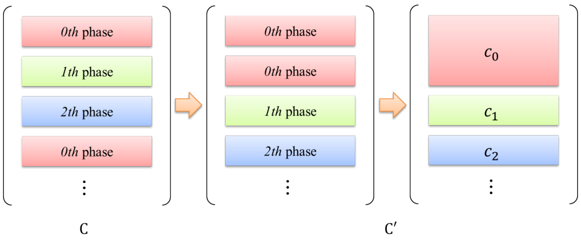


Figure 6: C' is the rearranged matrix from C. Each block matrix $\mathrm{c_i}$ contains discretized sensor data of ith phase.

5.5. Mapping with Sensor Data

After optimizing the low-dimensional latent variables X that represents the entire example motions Y, a mapping between the smartphone sensor data and the latent coordinate points is defined. Even in a low dimensional space, a linear mapping generally leads to poor results. We apply a non-linear neural network $\mathrm{P_{mlp}} = [\mathrm{p}_1, \ldots, p_m]^{\mathrm{T}}$, where $\mathrm{p_i}$ is a MLP model trained by each element of C and its corresponding elements in X in Section 5.4.

6. Motion Reconstruction

The goal is to closely estimate the 3D motion sequence of an actor, using sensor data from a smartphone. New input data are cleaned up by means of linear interpolation and a 1-D Gaussian filter, as described in Section 4.1. The result is saved as $\mathrm{S_{new}}$. Our method is robust against identical motions with different lengths due to the use of phase segmentation and is capable of smooth motion generation through interpolation in a latent space.

6.1. Phase Segmentation

Before finding the corresponding new latent points $\mathrm{X_{new}}$ from $\mathrm{S_{new}}$, phase classification of each frame is essential. We discretize values of $\mathrm{S_{new}}$ as an input sequence to the HMM built in Section 5.4. We use a window of size v $\mathrm{U} = [\mathrm{u}_1, \ldots, \mathrm{u}_{\mathrm{N}-\mathrm{v}+1}]^{\mathrm{T}}$. Figure 7 shows how majority voting works to obtain phase information for every frame. U moves frame by frame, and is applied to each dimension of the sensor data. For every window, the trained HMM determines the probability $\mathrm{P}\left(\mathrm{O}|U\right)$. The phase of U is determined as that with the highest value. Each frame is set to the phase through majority voting by the windows covering that frame.

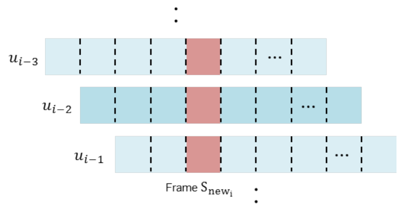


Figure 7: HMM classifies the phase of every window by selecting the phase with the highest probability, while shifting the window frame by frame. The phase of frame S_{new_i} is then determined by majority voting participated by all windows covering the frame S_{new_i}.

6.2. Pose Synthesis

Applying S_{new} to phase segmentation produces C_{new}. Plugging C_{new} into P_{mlp} constructed in Section 5.5, in turn, produces a mapped outcome of latent data X_{new}. The learned model $\Gamma = \{X, Y, \overrightarrow{\alpha}, \overrightarrow{\beta}, W\}$ from Section 5.3 can predict new pose data Y_{new} that correspond to the latent coordinate points by means of a Gaussian distribution.

$$y_{new} = WY^T K_Y^{-1} k(x_{new}) + bias \qquad (7)$$

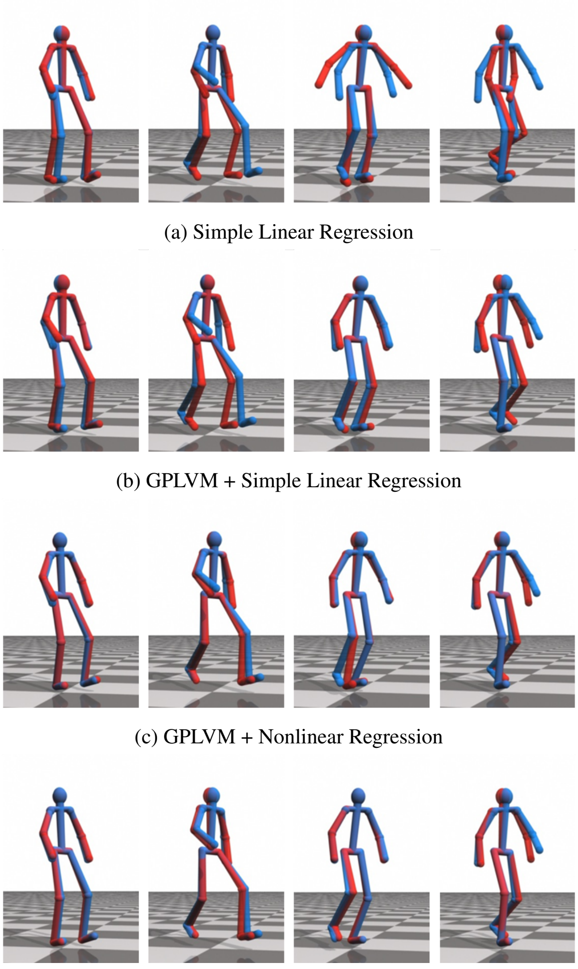
where $k(x)$ is an $N \times 1$ vector that is constructed from an *ith* element of $k_{rbf}(x_{new}, x_i)$, and x_{new} and y_{new} are components of X_{new} and Y_{new} for each frame. The bias term is a $d_y \times 1$ vector whose value is determined as the row-wise mean of the entire Y vectors used for the training.

As this operation is performed independently per frame, noise may be induced over time. We apply Laplacian smoothing to ensure temporal coherence for each angle value using the SLERP algorithm [Sho85] in a quaternion space. To avoid foot skating artifacts, we adjust the root translation on foot contact, which is automatically detected when the foot is sufficiently close to the ground and its velocity is below a threshold [KHKL09].

7. Results

To verify the effectiveness of our system, we trained the motion model using a two-step walking clip with 136 frames. We trained the HMM with a clip with 487 frames that contain a combination of 4 similar motions. We set the dimension to 5 for latent coordinates for the creation of the motion model. In the HMM, the size of the window was set to 10-20 according to the speed of motions, where a faster motion uses a smaller size window due to faster phase changes. The discretization level was set to 3.

We compare results from our system and ground-truth data obtained by using eight Vicon MX-40 120Hz 3D motion capture cameras. Each row of Figure 8 shows a comparison between results obtained after the application of each step of our method and ground truth data. The error of each method is visualized as the differences between the two colored characters. A direct mapping between full body pose vectors and the corresponding sensor data using a Simple Linear Regression produced poor results, as shown in the first row. The poor outcomes resulted from the mapping between very low and high dimensional data. The second row shows the results obtained by applying Simple Linear Regression between corresponding smartphone data and latent points derived from the GPLVM. Although the results are more stable than those derived from the direct mapping, detection of each pose is still unreliable. Applying MLP as nonlinear regression between the latent space and the sensor space produced the results in the third row. The leg motion became more realistic, but without temporal refinement, shaking artifacts were noticeable in some frames. The last



(a) Simple Linear Regression

(b) GPLVM + Simple Linear Regression

(c) GPLVM + Nonlinear Regression

(d) Our method

Figure 8: Comparison between results produced by applying each method (red) and ground truth data (blue).

row shows the final stabilized results produced by additionally incorporating state decomposition using a HMM for phase segmentation.

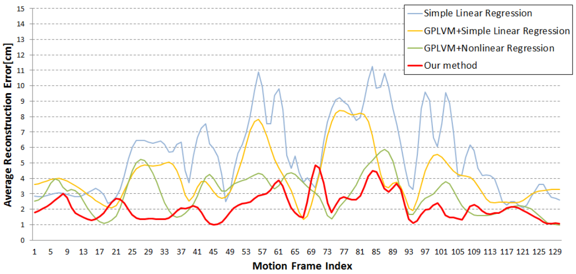


Figure 9: Reconstruction error generated by comparision with ground truth data.

The graph presented in Figure 9 shows the reconstruction error between ground truth data and results obtained after the application of each step of our method. As described in Tourneir et al. [TWC*09], producing a distance measure that matches human perception is inherently difficult. Therefore, we rely on the average Root Mean Square Error (RMSE) of all joint positions relative to the root. The average RMSE of our method is lower than 0.05 meter in every frame. This number is lower than the number reported in a previous study [TZK*11] that employed more than one sensor. The results from the method without a HMM (the green line in Figure 9) show better performance than those from our method in some frames. This is because our results are temporally much smoother due to the use of the HMM. Table 1 shows the average reconstruction error for different body parts. Interestingly, although the sensor was attached only on the leg, the magnitude of error of the joints in the upper body is not much different from that of the joints in the lower body. Note that while the error metric based on joint positions suffices to prove the superiority of our method, other distance metric like PCA metric [vBE09] can also be employed for more robust error measure.

	Full body	Upper body	Lower body
Reconstruction Error(cm)	1.51	1.52	1.39

Table 1: Average RMSE for body parts.

Figure 10 shows results of the reconstruction process for various motions. We generated a GPLVM model for each motion type. Note that the actors are different from the one for training data. It is clear that no special environmental constraints were imposed during the capture. There are differences in proportions between the simple stick figure and the human. Therefore, the angles of the joint may not match exactly. However, the overall characteristics of the motion are very similar between the two. In the clockwise direction from the top left, the images indicate straight walking, running, hopping, and jumping, respectively. The generated motions verify that our method is robust against variation of the types of locomotion and actor differences. Please refer to the accompanying video for more motion examples.

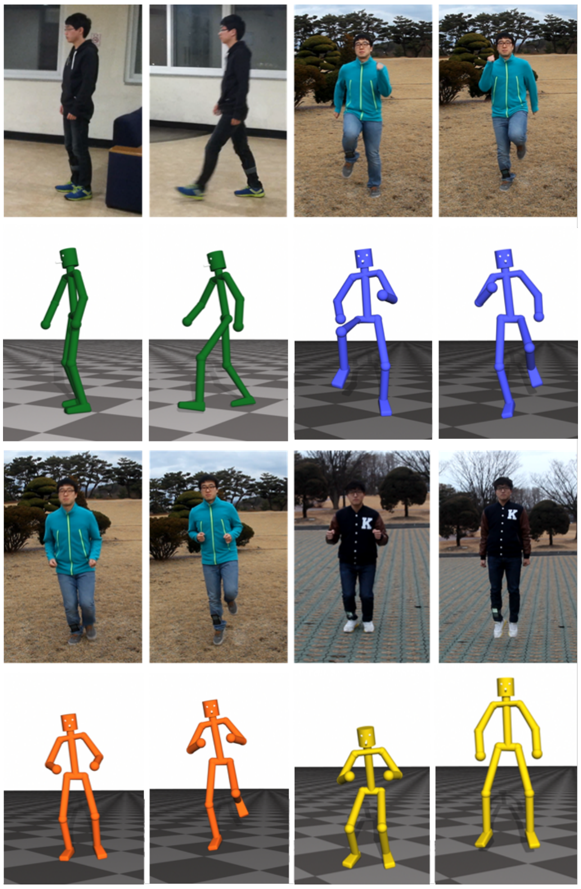


Figure 10: Results of the reconstruction process for various motions.

Motion	Number of frames		Computation time(sec)	
	Train	Test	Training	Reconstruction
Straight walking	135	123	40.363	0.001
Jumping	75	67	14.023	0.001
Running	74	84	13.971	0.001
Hopping	65	94	9.914	0.001

Table 2: Computation time.

For implementation, we utilized the MLPACK library [CCS*13] for HMM and ALGLIB [Boc01] for MLP. All of our simulations were performed on an Intel i7 3.4

GHz CPU with 8 GB of RAM. Table 2 shows the number of frames and the computational time required to produce the results. The preprocessing time to learn the motion model and to construct the state decomposition model increases linearly with the number of frames of a training motion sequence. Reconstructing motion for new sensor data, phase decomposition, and motion synthesis were achieved in less than a millisecond.

7.1. Disscusion and Limitations

We employed a smartphone as an inertial sensing device in our experiments, because most recent smartphones are equipped with reasonably accurate inertial sensors. In addition, sensor recording can be performed using the provided smartphone Software Development Kit that enables easy implementation. However, our method is not necessarily tied to the use of a smartphone. It can work equally well with any portable device containing inertial sensors such as an accelerometer and a gyroscope.

Because our system is aimed at solving a very challenging regression problem while an inertial sensor included in a smartphone does not produce very accurate data, it was not easy to generate motion details. Although we utilized velocity GP to produce improved accuracy regarding root translation, perfect turning motion was not generated as the horizontal root velocity could not be reconstructed faithfully. To obtain more stable root velocity, it is apparently necessary to consider additional degrees of sensor data around the horizontal plane or a proper feedback control stage after the motion reconstruction. Because of the attached sensor location, our reconstruction is focused on lower body motion. Therefore, upper body motions such as punching or shaking hands are difficult to reconstruct. We assumed that motions are captured in a natural manner. Therefore, the reproduction of specific artificial motions or partial actions is beyond the scope of this research.

Similar to other data-driven methods, motions that are not included in the motion capture database are impossible to recreate. Meanwhile, training with a large data set of motion can cause more ambiguity and result in bad poses. It is known that the performance of a GPLVM decreases as the size and heterogeneity of data set increase [CH05]. Therefore, finding a proper database size is important and this is a common problem with GPLVM-based methods. Our approach of learning the GPLVM for each phase motion separately may alleviate this situation to some degree. Reconstruction can be performed in less than a millisecond. However, the runtime process cannot be utilized online as the phase segmentation stage should be performed over entire sequences of sensor data.

8. Conclusion and Future Work

We present a novel approach to reconstruct full-body human locomotion using a single smartphone. Our method successfully maps low dimensional sensor data to full body 3D poses through the GPLVM and nonlinear regression. To consider temporal coherence, we applied a HMM and identified the phase of each frame automatically for new sensor input. As shown by various results, our method can reconstruct motions such that they are comparable to ground truth data.

Our technique can be further developed to capture more general motions such as turning and dancing. We will explore ways to operate our method online to handle new sensor input, via communication between a computer and a smartphone through Bluetooth or wireless internet. This real-time motion capture application will be useful for game control or crowd capture. As the required reconstruction time is usaully less than 1 ms, it may be possible to perform the reconstruction on mobile devices directly instead of connecting to a PC. Combining physics-based refinement is another possible way to create more natural motion.

9. Acknowledgements

We are grateful to Sungmin Jang, Jaedong Kim and Jaewon Song for data preparation and experiments. We also thank all the members of the Visual Media Laboratory. This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT and future Planning(2014R1A2A1A01002871).

References

[Alp04] ALPAYDIN E.: *Introduction to machine learning*. MIT press, 2004, pp. 305–326. 5

[BHG93] BADLER N. I., HOLLICK M. J., GRANIERI J. P.: Real-time control of a virtual human using minimal sensors. 2

[Boc01] BOCHKANOV S.: Alglib (www.alglib.net). 7

[CCS*13] CURTIN R. R., CLINE J. R., SLAGLE N. P., MARCH W. B., RAM P., MEHTA N. A., GRAY A. G.: MLPACK: A scalable C++ machine learning library. *Journal of Machine Learning Research 14* (2013), 801–805. 7

[CH05] CHAI J., HODGINS J. K.: Performance animation from low-dimensional control signals. In *ACM Transactions on Graphics (TOG)* (2005), vol. 24, ACM, pp. 686–696. 2, 8

[DYP03] DONTCHEVA M., YNGVE G., POPOVIĆ Z.: Layered acting for character animation. *ACM Transactions on Graphics (TOG) 22*, 3 (2003), 409–416. 2

[FTOK96] FREEMAN W. T., TANAKA K.-I., OHTA J., KYUMA K.: Computer vision for computer games. In *Automatic Face and Gesture Recognition, Proceedings of the Second International Conference on* (1996), IEEE, pp. 100–105. 2

[GMHP04] GROCHOW K., MARTIN S. L., HERTZMANN A., POPOVIĆ Z.: Style-based inverse kinematics. In *ACM Transactions on Graphics (TOG)* (2004), vol. 23, ACM, pp. 522–531. 2, 3

[KHKL09] KIM M., HYUN K., KIM J., LEE J.: Synchronized multi-character motion editing. In *ACM Transactions on Graphics (TOG)* (2009), vol. 28, ACM, p. 79. 6

[Kon01] KONAMI: Boxing and police 911 game, http://www.konami.com. 2

[Law04] LAWRENCE N. D.: Gaussian process latent variable models for visualisation of high dimensional data. *Advances in neural information processing systems 16*, 329-336 (2004), 3. 2

[Law05] LAWRENCE N.: Probabilistic non-linear principal component analysis with gaussian process latent variable models. *The Journal of Machine Learning Research 6* (2005), 1783–1816. 3

[LWH*12] LEVINE S., WANG J. M., HARAUX A., POPOVIĆ Z., KOLTUN V.: Continuous character control with low-dimensional embeddings. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 28. 2, 3, 4

[MCC09] MIN J., CHEN Y.-L., CHAI J.: Interactive generation of human animation with deformable motion models. *ACM Transactions on Graphics (TOG) 29*, 1 (2009), 9. 5

[OTH02] OORE S., TERZOPOULOS D., HINTON G.: A desktop input device and interface for interactive 3d character animation. In *Graphics Interface* (2002), vol. 2, pp. 133–140. 2

[PMBH*10] PONS-MOLL G., BAAK A., HELTEN T., MULLER M., SEIDEL H.-P., ROSENHAHN B.: Multisensor-fusion for 3d full-body human motion capture. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* (2010), IEEE, pp. 663–670. 2

[SH08a] SHIRATORI T., HODGINS J. K.: Accelerometer-based user interfaces for the control of a physically simulated character. In *ACM Transactions on Graphics (TOG)* (2008), vol. 27, ACM, p. 123. 2

[SH08b] SLYPER R., HODGINS J. K.: Action capture with accelerometers. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2008), Eurographics Association, pp. 193–199. 2

[Sho85] SHOEMAKE K.: Animating rotation with quaternion curves. *ACM SIGGRAPH computer graphics 19*, 3 (1985), 245–254. 6

[SHS98] SEMWAL S. K., HIGHTOWER R., STANSFIELD S.: Mapping algorithms for real-time control of an avatar using eight sensors. *Presence: Teleoperators and Virtual Environments 7*, 1 (1998), 1–21. 2

[Son03] SONY: Eyetoy systems, http://www.eyetoy.com. 2

[Sys02] SYSTEMS V. M.: http://www.vicon.com. 3

[TKZW] TAUTGES J., KRÜGER B., ZINKE A., WEBER A.: Tracking of human body motions using very few inertial sensors. 2

[TWC*09] TOURNIER M., WU X., COURTY N., ARNAUD E., REVERET L.: Motion compression using principal geodesics analysis. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 355–364. 7

[TZK*11] TAUTGES J., ZINKE A., KRÜGER B., BAUMANN J., WEBER A., HELTEN T., MÜLLER M., SEIDEL H.-P., EBERHARDT B.: Motion reconstruction using sparse accelerometer data. *ACM Transactions on Graphics (TOG) 30*, 3 (2011), 18. 2, 7

[UFG*08] URTASUN R., FLEET D. J., GEIGER A., POPOVIĆ J., DARRELL T. J., LAWRENCE N. D.: Topologically-constrained latent variable models. In *Proceedings of the 25th international conference on Machine learning* (2008), ACM, pp. 1080–1087. 2

[vBE09] VAN BASTEN B. J., EGGES A.: Evaluating distance metrics for animation blending. In *Proceedings of the 4th International Conference on Foundations of Digital Games* (2009), ACM, pp. 199–206. 7

[WFH08] WANG J. M., FLEET D. J., HERTZMANN A.: Gaussian process dynamical models for human motion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 30*, 2 (2008), 283–298. 2, 5

[YP03] YIN K., PAI D. K.: Footsee: an interactive animation system. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2003), Eurographics Association, pp. 329–338. 2