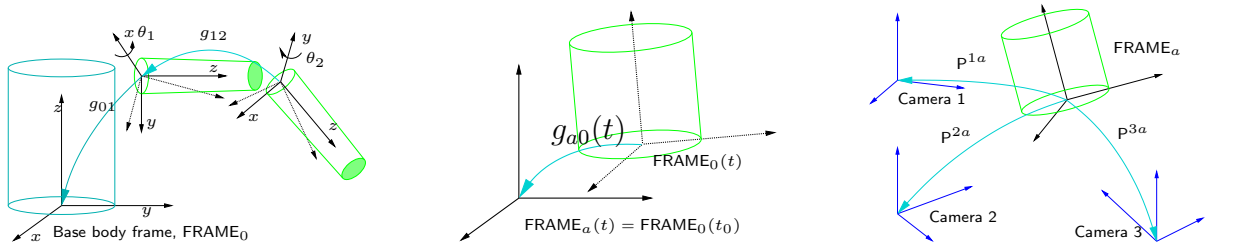
3D MODELLING OF HUMAN MOTION USING KINEMATIC CHAINS AND MULTIPLE CAMERAS FOR TRACKING

Aravind Sundaresan [1], Rama Chellappa [2] and Amit RoyChowdhury [3]
aravinds@cfar.umd.edu
[1,2] Center for Automation Research, University of Maryland, College Park, Maryland, USA
[3] Department of Electrical Engineering, University of California, Riverside, California, USA

INTRODUCTION

Simultaneous 3D human shape estimation and motion tracking is a very challenging problem. Given the complicated nature of human shape and motion, it can be very difficult to perform robust tracking of body parts without using a motion or shape model. Modelling the human body as rigid parts linked in a kinematic structure is a simple yet reasonably accurate model for tracking purposes. Optical flow can be exploited to provide dense information and obtain robust estimates of the motion parameters. Bregler and Malik (1998) use orthographic projection, while Sidenbladh et al. (2000) use a Bayesian formulation combined with a particle filtering approach to determine the motion parameters. Yamamoto et al. (1998) use a different set of motion parameters to perform tracking. They use multiple views and perspective projection in their model but have a large parameter set and make approximations in their formulation.

We use kinematic chains to model human motion using a perspective projection model in a multi-camera framework and arrive at a precise formulation for a tracker. We analyse the approximations we make in a practical system and provide an iterative algorithm for accurate estimation. We illustrate our model for human structure and motion in Figure 1 (a). We model the movement of human beings using kinematic chains with the root of the kinematic tree being the torso (base body). Each body part forearm, upper arm, torso, head, etc., is modelled as a rigid body, whose shape is known. At any given time we know the positions



(a) Structure of Chain (b) Motion of base at time t (c) Multiple camera projection

Figure 1: Kinematic Chain Schematic

and orientations of the objects in the kinematic chain. The instantaneous 3D velocity of each point in the base body coordinate system is a linear function of the vector of 3D motion parameters. In a computer vision system, however, we are able to measure only the 2D image velocities at each pixel for different cameras. Under perspective projection, the 2D velocities are linear in the 3D motion parameters. This enables us to accurately obtain the 3D motion parameters of the kinematic chain, given only the 2D pixel velocities obtained

from multiple cameras. In general, it is difficult to obtain the instantaneous pixel velocities ($\dot{\boldsymbol{x}}$) accurately because we work with a set of discrete-time images of the scene. Therefore, we have to work with pixel displacements ($\Delta\boldsymbol{x}$). The substitution of pixel velocity by pixel displacement introduces a bias in the estimation. We analyse the error introduced by this bias and propose an iterative algorithm by means of which we are able to decrease the bias.

OUTLINE OF METHODS

We can obtain 3D velocities of each pixel in terms of motion parameters of the body parts and then the 2D pixel velocities in image coordinates of each camera in terms of the 3D velocities in a common reference frame. We propose an algorithm that iteratively estimates the motion parameters, after taking into account the errors in estimation.

Points on the base body (attached to FRAME_0) are free to move with a rigid motion (rotation and translation) with respect to a static frame, FRAME_a. However, points on the i^{th} body part move in a constrained manner with respect to FRAME_0. The instantaneous velocity of this point, therefore, has two components: $\boldsymbol{v}_B(t)$, due to the motion of the base body, and $\boldsymbol{v}_K(t)$ due to the motion of the kinematic chain.

We can show that $\boldsymbol{v}_B(t) = \mathsf{W}(\boldsymbol{q}_a)V_{a0}^s$ and $\boldsymbol{v}_K(t) = \mathsf{E}(\boldsymbol{q}_a, \boldsymbol{\theta})\dot{\boldsymbol{\theta}}$, where W and E are known matrix functions of $\boldsymbol{q}_a$ and $\boldsymbol{\theta}$. V_{a0}^s comprises the translational velocity $\boldsymbol{v}_{a0}^s$ and the rotational velocity $\boldsymbol{\omega}_{a0}^s$ of the base body. $\boldsymbol{\theta}$ describes the "configuration" of the kinematic chain (each element parameterises the orientation between two connected components in the kinematic chain in terms of the angle between the reference frame axes) and $\dot{\boldsymbol{\theta}}$ is its derivative. We see that the complete 3D motion parameters are given by vectors V_{a0} and $\dot{\boldsymbol{\theta}}$. Combining both, we get $\dot{\boldsymbol{q}}_a(t) = \mathsf{M}(\boldsymbol{q}_a, \boldsymbol{\theta})\boldsymbol{\varphi}$ where $\mathsf{M}(\boldsymbol{q}_a, \boldsymbol{\theta})$ is a matrix function of the current pose of the base body and configuration of the kinematic chain and $\boldsymbol{\varphi} = \begin{bmatrix} V_{a0} \\ \dot{\boldsymbol{\theta}} \end{bmatrix}$.

We can show that the instantaneous image velocity of a point, under perspective projection, is also linear in the motion parameters. Assume there are C cameras, numbered from 1 through C. Let $\mathsf{P}^{ca} = [\boldsymbol{p}_1^{ca}, \boldsymbol{p}_2^{ca}, \boldsymbol{p}_3^{ca}]^\mathsf{T}$ map a point in FRAME_a, $\boldsymbol{q}_a = [X_a, Y_a, Z_a, 1]^\mathsf{T}$ to the homogeneous image coordinates, $\tilde{\boldsymbol{q}}_c = [\tilde{x}_c, \tilde{y}_c, \tilde{z}_c]^\mathsf{T}$, of the c^{th} camera as in Figure 1 (c). We then derive the 2D velocity of a point, $\boldsymbol{u}_c$ in terms of the 3D velocity $\dot{\boldsymbol{q}}$ as $\boldsymbol{u}_c = \mathsf{C}^c(\boldsymbol{q}_a, \mathsf{P}^{ca})\dot{\boldsymbol{q}}$.

Given that K is the number of degrees of freedom in the kinematic chain and $M = K+6$, we have the linear relation for a single pixel in camera c: $\boldsymbol{u}_c = \mathsf{C}^c_{2\times 3}(\boldsymbol{q}_a, \mathsf{P}^{ca})\mathsf{M}_{3\times M}(\boldsymbol{q}_a, \boldsymbol{\theta})\boldsymbol{\varphi}_{M\times 1}$. We can estimate $\boldsymbol{\varphi}$ using the above set of equations for all pixels and all cameras.

Dropping subscripts and simplifying we get $\boldsymbol{u} = \mathsf{A}(t)\boldsymbol{\varphi}$. where $\mathsf{A}(t) = \mathsf{C}^c(\boldsymbol{q}_a, \mathsf{P}^{ca})\mathsf{M}(\boldsymbol{q}_a, \boldsymbol{\theta})$. In general we work with pixel displacement, $\Delta\boldsymbol{x}$, rather than pixel velocity $\boldsymbol{u} = \dot{\boldsymbol{x}}$.[1] We, therefore, try to formulate our estimator accordingly and analyse the error introduced by the approximation. We can expand the pixel position in terms of the Taylor series: $\boldsymbol{x}(t+\Delta t) = \boldsymbol{x}(t) + \dot{\boldsymbol{x}}(t)\Delta t + \mathcal{O}(\Delta t^2)$ and we get $\Delta\boldsymbol{x} \triangleq \boldsymbol{x}(t+\Delta t) - \boldsymbol{x}(t) = \dot{\boldsymbol{x}}(t)\Delta t + \mathcal{O}(\Delta t^2)$. We drop the dependence on time in the following equations. We model the measured pixel displacement,

[1] Since we have only a discrete set of images of the scene, estimation of arbitrary instantaneous velocities will be affected by the time-discretisation leading to an error in the estimates.

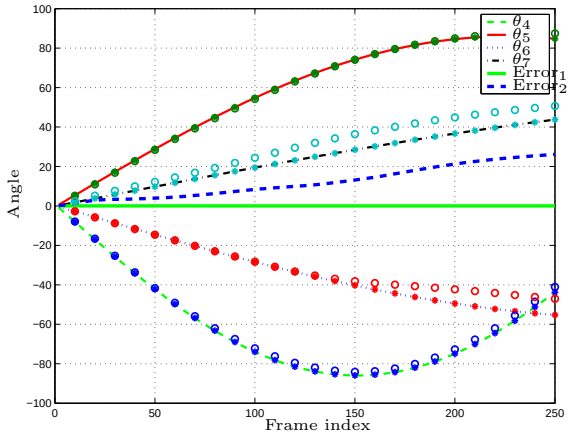


Figure 2: Comparison of Iterative and Non-Iterative Algorithms.

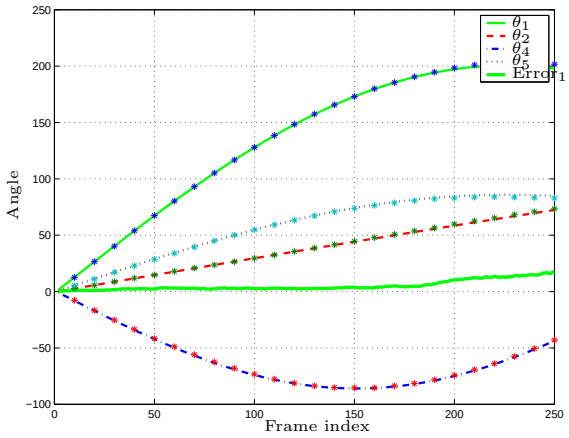


Figure 3: Kinematic chain angle estimation

$\Delta\tilde{\boldsymbol{x}}$, as $\Delta\tilde{\boldsymbol{x}} = \Delta\boldsymbol{x} + \boldsymbol{\eta}$, where $\boldsymbol{\eta}$ is zero-mean measurement noise with variance Σ. We therefore have the following equation.

$$\Delta\tilde{\boldsymbol{x}} = \dot{\boldsymbol{x}}(t)\Delta t + \mathcal{O}(\Delta t^2) + \boldsymbol{\eta} = \mathsf{A}\boldsymbol{\varphi}\Delta t + \mathcal{O}(\Delta t^2) + \boldsymbol{\eta} \tag{1}$$

If we ignore second and higher order terms, then we have our estimator $\Delta\tilde{\boldsymbol{x}} = \mathsf{A}\boldsymbol{\varphi}\Delta t + \boldsymbol{\eta}$. The higher order terms can be interpreted as a bias that affects the estimator. We propose an iterative algorithm to eliminate this bias. In our iterative algorithm, we first estimate the parameters using the linear equation. We then compute the 2D motion (pixel displacements) for the estimated motion. The computation of 2D motion, given the motion parameters is exact. We then set the new pixel displacements to be the difference between the observed pixel displacements and the computed pixel displacements and iterate. As we thus get closer to the true solution, the linear estimation becomes more accurate and we ultimately arrive at an accurate estimate of the true motion parameters.

EXPERIMENTAL RESULTS AND DISCUSSION

In our experiments we have determined how our estimator for 3D motion parameters performs on simulated data (with and without noise). We have built a kinematic chain composed of a base body with three articulated objects attached in a kinematic chain. The base body moves with translational and rotational motion and the objects are constrained to rotate about their joints with varying degrees of freedom. Results (tracking of some of the motion parameters) using our algorithm in the iterative and non-iterative modes on the simulated data are in Figure 2. The solid lines are the true values of the parameters, the "$*$" markers are the outputs of the iterative tracker, and the "$\circ$" markers are the outputs of the non-iterative tracker. The errors in the tracking for Algorithm 1 (iterative algorithm) and Algorithm 2 (non-iterative) are also plotted. We see that the error in the iterative tracker (Error 1) is almost zero while the error in the non-iterative (Error 2) tracker is substantial. We find that the iterative tracker is much superior and results in an almost perfect tracker on data that

does not contain noise. We also present in Figure 3, the performance of the iterative tracker on simulated data to which zero-mean Gaussian measurement noise has been added. The noise has a standard deviation ($\sigma = 0.0006$) that is 20% of the average pixel displacement (0.0030). The average relative error in the joint angles after 50 frames is less than 0.01. However, due to the incremental nature of the estimation, the error accumulates with time.

We would like to refine the estimate using independent information from the images such as the image edge maps. We are currently working on modifying the algorithm to include the body part edge information to refine our estimates. We are also working on incorporating a more general and accurate 3D model of the human being tracked (using 3D scanner data). We can also use stochastic models for human motion leading to more robust estimation.

We are currently applying our algorithm on real images collected using multiple cameras in the KECK laboratory at the University of Maryland, College Park. The details of the experiments and the KECK laboratory can be found at `http://www.cfar.umd.edu/~aravinds/kecklab.html` and `http://www.cfar.umd.edu/~aravinds/kecklab.html` respectively. We are working with eight cameras positioned approximately at equal distances on a circumference of a circle. We would like to capture human motion from a 360 degree perspective, perform complete calibration for the multi-camera rig, and use appropriate optical flow techniques to obtain pixel displacements. We will provide initial pose in the world reference frame manually and track the human body motion using video sequences obtained from multiple cameras.

SUMMARY

In this paper, we have outlined an algorithm for estimating 3D motion parameters of a kinematic chain model from 2D image pixel displacements using a kinematic chain model for human motion and use the perspective camera projection model and multiple cameras. We also report results on simulated data that contains measurement noise and find that the iterative algorithm is still robust and successfully tracks the motion parameters. The compact form of the 3D motion parameter vector, and the linear formulation make the algorithm robust.

REFERENCES

Bregler, C. and Malik, J. (1998). Tracking people with twists and exponential maps. In *Computer Vision and Pattern Recognition*, pages 8–15.

Murray, R. M., Li, Z., and Sastry, S. S. (1994). *A Mathematical Introduction to Robotic Manipulations*. CRC Press.

Sidenbladh, H., Black, M. J., and Fleet, D. J. (2000). Stochastic tracking of 3D human figures using 2D image motion. In *European Conference on Computer Vision*, pages 702–718.

Yamamoto, M., Sato, A., Kawada, S., Kondo, T., and Osaki, Y. (1998). Incremental tracking of human actions from multiple views. In *Computer Vision and Pattern Recognition*, pages 2–7.