G-Flow: A Generative Model for Fast Tracking Using 3D Deformable Models

Javier R. Movellan, John Hershey, Tim K. Marks, Cooper Roddey
(MPLab TR 2003.03 v2)
Machine Perception Laboratory Institute for Neural Computation
University of California San Diego
La Jolla, CA 92093-0515

Abstract

*We present a generative model (*G-flow*) and inference algorithm for simultaneous tracking of 3D pose, non-rigid motion, object texture and background texture. Under this model optimal inference about pose and texture can be performed efficiently using a bank of Kalman filters for texture whose parameters are updated by an optic-flow-like algorithm. The inference algorithm unifies optic flow-based and texture-based tracking methods, dynamically adjusting the relative importance of each component in a principled manner. Classic optic flow and template-based algorithms emerge as special cases, and the conditions under which they are optimal are elucidated by the model. For instance, the Lucas-Kanade optic-flow algorithm is a special case that is optimal under certain conditions (complete certainty of the current location of the object in each frame, and knowledge of its texture only via its current location).*

1 Introduction

Many approaches have been proposed in the computer vision literature to solve the object tracking problem. In general these can be divided into motion-based and template-based approaches. Motion-based approaches compute local estimates of optic flow, typically using a variation of the Lucas-Kanade optic-flow algorithm [5], then combine these estimates using global object constraints [2]. The advantage of a motion-based approach is that it makes few assumptions about the appearance of the object being tracked. When given two images y_t, y_{t+1} at two consecutive time steps, and the position of the object at time t, the approach gives us an estimate of the position of the object at time $t+1$. This method implicitly assumes good knowledge about the location of the object at each time step, and thus it has a tendency to drift as errors accumulate. Initialization and recovery from drift are open issues in motion-based approaches, and they are typically handled using heuristic methods.

At the other end of the spectrum template approaches assume good knowledge about the appearance of the object of interest. The advantage of these approaches is that they require little knowledge about the current location of the object, provided the template is correct. Local or global search methods are then used to find the pose that best fits the image plane. A known problem with template-based approaches is dealing with realistic sources of variation (pose, illumination, identity, expression, etc). Template-based methods typically rely on heuristics that allow for dynamic updating of the templates and periodic re-registration.

In practice, the issues of model initialization, dynamic update of templates, error detection, and re-initialization are still unsolved. Finding principled solutions to these problems is arguably the most important impediment to the widespread application of computer vision technology in daily life.

In this paper, we present a generative model (G-flow) for video sequences. The model, while relatively simple, provides a rich framework for analyzing the problem of how to dynamically combine motion-based and texture-based information in an optimal manner. A contribution of the model is that classic optic flow and template-based algorithms emerge as special cases of optimal inference under limited conditions. Optic flow is optimal when the location of the object is known and its appearance is unknown. Template-based algorithms are optimal in the opposite case. In practice optimal inference under G-flow comprises a combination of motion and template-based information that is dynamically re-weighted as new images are presented. Standard approximations can be used to solve the inference problem very quickly, allowing for on line, real time 3D pose and expression tracking, geometry estimation, and texture recovery.

2 Video generation model

Unless otherwise stated capital letters represent random variables, small letters represent specific values taken by random variables, and Greek letters represent fixed model parameters. When possible we use informal shorthand notation and identify probability functions by their arguments. We also drop commas between arguments in probability functions. For example, $p(xy)$ is shorthand for the probability (or probability density) that the random variable X takes the specific value x and the random variable Y takes the

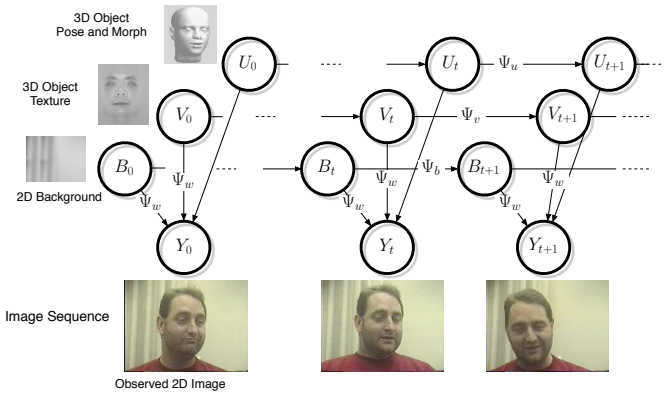

Figure 1: *The G-flow video generation model: The pose and texture of the object live in 3D are projected onto 2D and then combined with the background to generate the observed video sequence. The model parameters include the initial distributions π_u, π_v, π_b, the texture transition certainties $\Psi_v \Psi_b$, the rendering noise parameter Ψ_w and the pose transition probabilities $p(u_t \,|\, u_{t-1})$. Except for the pose transition probabilities, the distributions controlled by these parameters are assumed Gaussian. The goal is to make inferences about (U_t, V_t, B_t) based on the observed video sequence $Y_1 \cdots Y_t$.*

value y. We use subscripted columns to indicate sequences. For example $X_{1:t} = X_1 \cdots X_t$. The term I_p stands for a $p \times p$ unit matrix. E stands for expected value, Var for covariance matrix and Pcs for precision matrix, the inverse of the covariance matrix. $A^n \otimes A^c$ refers to the set of $r \times c$ matrices whose cells are elements of A. The following terms will be used throughout the paper:

- $y_t \in \mathbb{R}^p$, the vectorized version of an image with p pixels.
- $u_t \in \mathbb{R}^{2n}$ a vector containing the position of n points on the image plane. These n points are thought to belong to the same object, the rest of the points on the image plane belong to the background.
- $v_t \in \mathbb{R}^n$, $b_t \in \mathbb{R}^p$ vectors with the texture map of the object and background respectively. We refer to each element of v_t and b_t as a *texel*.
- $a_v : \mathbb{R}^n \to \{0,1\}^p \otimes \{0,1\}^n$, a function whose input is the position of the object points on the image plane and whose output is a $p \times n$ matrix of zeroes and ones. If there is a one at row i, column j it means that the j^{th} object point projects on pixel i. There should be a total of n ones and at most a one per row.
- $a_b : \mathbb{R}^n \to \{0,1\}^p \otimes \{0,1\}^p$ is a function whose input is the position of the object points on the image plane and whose output is a $p \times p$ diagonal matrix. If there is a one at row j, column j it means that the background texel j projects on pixel j on the image plane. We put the constraint that if $a_{vij} = 1$ then $a_{bjj} = 0$, i.e., if a pixel i is rendered by the object, it is not rendered by the background.

The functions a_v, a_b encapsulate the projection model and filtering effects of the imaging device.

Example: Suppose we have a 4-pixel image plane, $p = 4$, and a 2-point object $n = 2$. Suppose the object can only take 2 locations in 3D: $q_1 = (-1, 0, 1)$ $q_2 = (1, 0, 1)$. When at q_1 the object projects onto the two pixels on the left. When at q_2 it projects on the two pixels on the right.

$$a_v(q_1) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} a_v(q_2) = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$a_b(q_1) = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} a_b(q_2) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Model Specification: G-flow models the video sequence as a stochastic process governed by a partially observable difference equation (see Figure 1 and 2). There are three hidden processes: A background process B, an object motion process U, and an object texture process V. They generate images as follows: The object pose, U_t determines which pixels the object and background project on, which we formulate using the projection function $c(U_t) = \big(a_v(U_t), a_b(U_t)\big)$. The object and background textures V_t and B_t then project to the image Y_t via $c(U_t)$ with additive noise as formulated below:

$$Y_t = c(U_t) \begin{pmatrix} V_t \\ B_t \end{pmatrix} + W_t, \quad \text{for } t = 1, \cdots \tag{1}$$

The system dynamics are as follows:

$$\begin{array}{ll} U_t \sim p(u_t \mid u_{t-1}) & \text{for } t = 2, \cdots \\ V_t = V_{t-1} + Z^v_{t-1} & \text{for } t = 2, \cdots \\ B_t = B_{t-1} + Z^b_{t-1} & \text{for } t = 2, \cdots \end{array}$$

$p(u_t \mid u_{t-1})$ is the pose transition distribution, Z^v, Z^b, W are sequences of zero mean, Gaussian processes independent of each other and of the initial conditions. Their respective precision matrices are Ψ_v, Ψ_b, Ψ_w. The form of the pose distribution is left unspecified for the sake of generality. Because the image generation process is nonlinear as a function of pose, our methods must accommodate this nonlinearity anyway, and hence we need not restrict the motion dynamics to a Gaussian form.

The model is specified by the following terms: (1) Initial conditions, which consist of a distribution for the object position U_1, and Gaussian distribution of object and background texture, V_1 and B_1, all of which are independent of each other. In addition we assume the

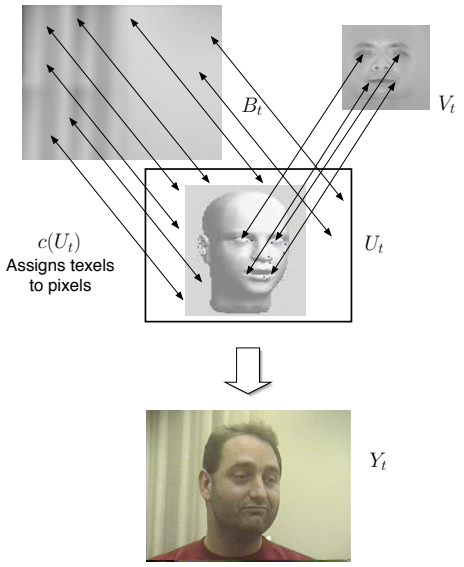

Figure 2: $c(U_t)$ *determines which texel is responsible for rendering each pixel on the image plane. Some of these will be rendered by object texels, some by background texels.*

variance of V_1 is diagonal and the variance of B_1 is a scalar times a unit matrix. (2) The precision matrices for the state transitions, Ψ_v, Ψ_b. (3) The pose transition distribution $p(u_t \mid u_{t-1})$. (4) The precision matrix for the image rendering noise is of the form $\Psi_w = I_p \sigma_w^{-1}$, where σ_w is a scalar. The imaging model (e.g., perspective projection) determines the functions a_v and a_b.

Structure of the Inference Problem: Inference requires computing the distribution of pose and texture given an observed sequence of images. The main difficulty in solving this problem centers around the motion posterior U_t. Since the object and background textures are not a linear function of the position of the pixel, then the observed images Y_t will in general not be a linear function of U_t. However, if $U_{1:t}$ were known then the object and background texture processes V_t, B_t would be linear and Gaussian and thus could be solved using Kalman filter equations with time variant parameters, as determined by $U_{1:t}$. This suggests the following scheme: Use approximate methods to obtain highly probable samples, of $U_{1:t}$, then use Kalman filtering equations to determine the distribution of $V_{1:t}B_{1:t}$ for each sample. Another important aspect of the problem, that we want to use to our advantage, is that the observed images have a strong spatio-temporal structure.

3 Filtering Distribution

Our goal is to find an expression for the filtering distribution $p(u_t v_t b_t \mid y_{1:t})$, for $t = 0, \cdots$. Using the law of total probability we have that

$$p(u_t v_t b_t \mid y_{1:t}) = \int p(u_t v_t b_t u_{1:t-1} \mid y_{1:t}) du_{1:t-1} \quad (2)$$

$$= \int p(u_t v_t b_t \mid u_{1:t-1} y_{1:t}) p(u_{1:t-1} \mid y_{1:t}) du_{1:t-1} \quad (3)$$

We can think of the first term $p(u_t v_t b_t \mid u_{1:t-1} y_{1:t})$ as *the opinion* about u_t, v_t, b_t of an expert that believes in the past the object was at $u_{1:t-1}$. The second term of the equation $p(u_{1:t-1} \mid y_{1:t})$ is the *credibility* of that expert.

3.1 The Opinion Equations

We decompose the opinion of expert $u_{1:t-1}$, into the product of the opinion about pose U_t times the opinion about texture V_t, B_t given pose.

$$p(u_t v_t b_t \mid u_{1:t-1} y_{1:t}) = p(v_t b_t \mid u_{1:t} y_{1:t}) p(u_t \mid u_{1:t-1} y_{1:t}) \quad (4)$$

Texture opinions: Because V_1, B_1 are Gaussian, the distribution of $V_t B_t$ given $u_{1:t-1} y_{1:t-1}$ is also Gaussian with a mean and covariance that can be obtained using time dependent Kalman estimation equations (a.k.a. the correction equations)

$$\begin{aligned} &Pcs(V_t B_t \mid u_{1:t} y_{1:t}) = \\ &Pcs(V_t B_t \mid u_{1:t-1} y_{1:t-1}) \\ &+ c(u_t)' \Psi_w c(u_t) \end{aligned} \quad (5)$$

$$\begin{aligned} &E(V_t B_t \mid u_{1:t} y_{1:t}) = \\ &Var(V_t B_t \mid u_{1:t} y_{1:t}) \end{aligned} \quad (6)$$

$$\begin{aligned} &[Pcs(V_t B_t \mid u_{1:t-1} y_{1:t-1}) \\ &E(V_t B_t \mid u_{1:t-1} y_{1:t-1}) \\ &\quad + c(u_{t-1})' \Psi_w y_{t-1}] \end{aligned} \quad (7)$$

This requires the distribution of $V_t B_t$ given $u_{1:t-1} y_{1:t-1}$, which can be obtained using the Kalman prediction equations

$$\begin{aligned} &E(V_t B_t \mid u_{1:t-1} y_{1:t-1}) = E(V_{t-1} B_{t-1} \mid u_{1:t-1} y_{1:t-1}) \\ &Var(V_t B_t \mid u_{1:t-1} y_{1:t-1}) = Var(V_{t-1} B_{t-1} \mid u_{1:t-1} y_{1:t-1}) \\ &+ \begin{pmatrix} \Psi_v^{-1} & 0 \\ 0 & \Psi_b^{-1} \end{pmatrix} \end{aligned} \quad (8)$$

Note the expected value $E(V_t B_t \mid u_{1:t} y_{1:t})$ contains texture maps (templates) for the object and background. $Var(V_t B_t \mid u_{1:t} y_{1:t})$ keeps the degree of uncertainty about the object and background templates. Due to the fact that pixels cannot be simultaneously rendered by the object and background, i.e., $a_{vij}(u_t) = 1 \rightarrow a_{bjj}(u_t) = 0$, and a_v is a permutation matrix, and a_b is diagonal, it can be shown that $Var(V_t B_t \mid u_{1:t} y_{1:t})$ has the same structure as $Var(V_0 B_0)$, i.e., it is diagonal, and the variances of all the B_t elements given $u_{1:t} y_{1:t}$ are equal.

Pose Opinions: The projection function $c(u_t)$ determines how the object and background templates render the image plane, i.e., which pixels are rendered by the object and which are rendered by the background. Since the effect of u_t on the likelihood function is nonlinear, we will not attempt to find an analytical solution for the pose opinion equations. Instead we will find the most probable value of u_t, given $u_{1:t-1}y_{1:t}$ for each expert and approximate the distribution as a Gaussian bump about that point. Note

$$p(u_t \mid u_{1:t-1}y_{1:t}) = \frac{p(y_{1:t-1} \mid u_{1:t-1})}{p(y_{1:t} \mid u_{1:t-1})}\, p(u_t \mid u_{t-1}) \\ p(y_t \mid u_{1:t}y_{1:t-1}) \tag{9}$$

where

$$p(y_t \mid u_{1:t}y_{1:t-1}) = \\ \int p(v_t b_t \mid u_{1:t-1}y_{1:t-1}) p(y_t \mid u_t v_t b_t) dv_t db_t \tag{10}$$

using the fact that V_t, B_t are independent of U_t given $u_{1:t-1}y_{1:t-1}$, i.e.,

$$\begin{aligned} p(u_t v_t b_t \mid u_{1:t-1}y_{1:t-1}) &= \int p(v_{t-1}b_{t-1} \mid u_{1:t-1}y_{1:t-1}) \\ & \quad p(u_t v_t b_t \mid u_{1:t-1} v_{t-1} b_{t-1}) dv_{t-1} db_{t-1} \\ &= \int p(v_{t-1}b_{t-1} \mid u_{1:t-1}y_{1:t-1}) p(u_t \mid u_{t-1}) \\ & \quad p(v_t b_t \mid v_{t-1} b_{t-1}) dv_{t-1} db_{t-1} \\ &= p(u_t \mid u_{1:t-1}y_{1:t-1}) \int p(v_{t-1}b_{t-1} \mid u_{1:t-1}y_{1:t-1}) \\ & \quad p(v_t b_t \mid v_{t-1} b_{t-1} u_{1:t-1} y_{1:t-1}) dv_{t-1} db_{t-1} \\ &= p(u_t \mid u_{1:t-1}y_{1:t-1}) p(v_t b_t \mid u_{1:t-1}y_{1:t-1}) \end{aligned} \tag{11}$$

We saw in the previous section that $p(v_t b_t | u_{1:t-1}y_{1:t-1})$ is Gaussian. Since $p(y_t \mid u_t v_t b_t)$ is also Gaussian it follows that $p(y_t | u_{1:t}y_{1:t-1})$ is Gaussian with the following mean and variance:

$$E(Y_t \mid u_{1:t}y_{1:t-1}) = c(u_t) E(V_t B_t \mid u_{1:t-1}y_{1:t-1}) \tag{12}$$

$$Var(Y_t \mid u_{1:t}y_{1:t-1}) = \Psi_w^{-1} \\ + c(u_t) Var(V_t B_t \mid u_{1:t-1}y_{1:t-1}) c(u_t)' \tag{13}$$

Let $\mathcal{O}(u_t)$ be an ordered set of indices to the pixels rendered by the object according to u_t. For $i \in \mathcal{O}(u_t)$ let $\mu_v(u_{1:t}, i)$ be the texel from the object texture map $E(V_t \mid u_{1:t-1}y_{1:t-1})$, that renders the image pixel i as determined by u_t. Let $\sigma_v(u_{1:t}, i)$ be the variance of that texel. For $j \notin \mathcal{O}(u_t)$ let $\mu_b(u_{1:t}, j)$ be the texel from the background texture map $E(B_t \mid u_{1:t-1}y_{1:t-1})$, that renders the image pixel j as determined by u_t, and let $\sigma_b(u_{1:t}, j)$ the variance of that texel. It follows that

$$\begin{aligned} \log p(y_t \mid u_{1:t}y_{1:t-1}) &= -\frac{1}{2} \log |Var(Y_t \mid u_{1:t}, y_{1:t-1})| \\ & - \frac{1}{2} \sum_{i \in \mathcal{O}(u_t)} \frac{(y_t(i) - \mu_v(u_{1:t}, i))^2}{\sigma_v(u_{1:t}, i) + \sigma_w} \\ & - \frac{1}{2} \sum_{j \notin \mathcal{O}(u_t)} \frac{(y_t(i) - \mu_b(u_{1:t}, i))^2}{\sigma_b(u_{1:t}, i) + \sigma_w} \end{aligned} \tag{14}$$

Moreover u_t simply permutes $Var(Y_t \mid u_{1:t}y_{1:t-1})$ and $E(Y_t | u_{1:t}y_{1:t-1})$. Thus $|Var(Y_t | u_{1:t}y_{1:t-1})|$ is constant with respect to u_t. Let

$$\hat{u}_t(u_{1:t-1}) = \underset{u_t}{\text{argmax}}\ p(u_t \mid u_{1:t-1}y_{1:t}) \tag{15}$$

Thus

$$\begin{aligned} \hat{u}_t(u_{1:t-1}) &= \underset{u_t}{\text{argmax}} p(u_t \mid u_{t-1}) p(y_t \mid u_{1:t}y_{1:t-1}) \\ &= \underset{u_t}{\text{argmin}} \frac{1}{2} \sum_{i \in \mathcal{O}(u_t)} \frac{(y_t(i) - \mu_v(u_{1:t}, i))^2}{\sigma_v(u_{1:t}, i) + \sigma_w} \\ & + \frac{1}{2} \sum_{j \notin \mathcal{O}(u_t)} \frac{(y_t(i) - \mu_b(u_{1:t}, i))^2}{\sigma_b(u_{1:t}, i) + \sigma_w} - \log p(u_t \mid u_{t-1}) \end{aligned}$$

Moreover, since

$$\sum_{j \notin \mathcal{O}(u_t)} \frac{(y_t(i) - \mu_b(u_{1:t}, i))^2}{\sigma_b(u_{1:t}, i) + \sigma_w} = \sum_j \frac{(y_t(i) - \mu_b(u_{1:t}, i))^2}{\sigma_b(u_{1:t}, i) + \sigma_w} \\ - \sum_{j \in \mathcal{O}(u_t)} \frac{(y_t(i) - \mu_b(u_{1:t}, i))^2}{\sigma_b(u_{1:t}, i) + \sigma_w} \tag{16}$$

and $\sum_j \frac{(y_t(i) - \mu_b(u_{1:t}, i))^2}{\sigma_b(u_{1:t}, i) + \sigma_w}$ is constant with respect to u_t, it follows that

$$\hat{u}_t(u_{1:t-1}) = \underset{u_t}{\text{argmin}} \frac{1}{2} \sum_{i \in \mathcal{O}(u_t)} \left(\frac{(y_t(i) - \mu_v(u_{1:t}, i))^2}{\sigma_v(u_{1:t}, i) + \sigma_w} \right. \\ \left. - \frac{(y_t(i) - \mu_b(u_{1:t}, i))^2}{\sigma_b(u_{1:t}, i) + \sigma_w} \right) - \log p(u_t \mid u_{t-1}) \tag{17}$$

$\hat{u}_t(u_{1:t-1})$ can be found very quickly using a Gauss-Newton method. The inverse Hessian $\hat{\sigma}_t(u_{1:t-1})$ also falls out easily from the Gauss-Newton method. The posterior distribution can then be approximated as a Gaussian $g(\cdot \mid \hat{u}_t(u_{1:t-1}), \hat{\sigma}_t(u_{1:t-1}))$ centered at $\hat{u}_t(u_{1:t-1})$ and with variance $\hat{\sigma}_t(u_{1:t-1})$.

Optic Flow as a Special Case: Suppose $p(u_t | u_{t+1})$ is uninformative, the background is a white noise process, i.e. $\sigma_b(u_t, i) \to \infty$ for all t, i and by time $t-2$ we are completely uncertain about the object texture, i.e.

$$Var(V_{t-1} \mid u_{1:t-2}y_{1:t-2}) \to \infty \tag{18}$$

It follows that

$$E(V_t \mid u_{1:t-1}y_{1:t-1}) = a_v(u_{t-1})y_{t-1} \tag{19}$$

i.e, our object texture map at time t is determined by the pixels from y_{t-1} that according to u_{t-1} are rendered by the object. Thus

$$\begin{aligned}&\operatorname*{argmax}_{u_t}\ p(u_t \mid u_{t-1}y_{1:t}) = \\ &= \operatorname*{argmin}_{u_t} \sum_{i\in\mathcal{O}(u_t)} \frac{\left(y_t(i) - a_v(u_{t-1})y_{t-1}(i)\right)^2}{\sigma_v(u_t,i) + \sigma_w}\end{aligned} \tag{20}$$

The most probable u_t is that which minimize the mismatch between the image pixels rendered by the object at time $t-1$ and the image at y_t shifted according to u_t. The Lucas-Kanade optic flow algorithm is simply the Newton-Gauss method as applied to minimize this error function.

Template matching as a Special Case: If $p(u_t \mid u_{t-1})$ is uninformative, the background is a white noise process and by time $t-2$ we are certain about the object texture map, i.e., $Var(V_{t-1} \mid u_{1:t-2}y_{1:t-2}) = 0$, then

$$E(V_t \mid u_{1:t-1}y_{1:t-1}) = E(V_t \mid u_{1:t-2}y_{1:t-2}) \tag{21}$$

and

$$\begin{aligned}&\operatorname*{argmax}_{u_t}\ p(u_t \mid u_{t-1}y_{1:t}) = \\ &= \operatorname*{argmin}_{u_t} \sum_{i\in\mathcal{O}(u_t)} \frac{\left(y_t(i) - \mu_v(u_t,i)\right)^2}{\sigma_w}\end{aligned} \tag{22}$$

where $\mu_v(u_t, i)$ is simply the fixed object template, shifted by u_t. This is the error function minimized by standard template match algorithms.

General Case: In general minimizing (17) results in a weighted sum of optic flow and template matching, with the weight of each approach depending on the certainty about the object template.

Importance Sampling: Suppose we are given a set of pose sequences $\{u^{(i)}_{1:t-1} : i = 1 \cdots n_{t-1}\}$. For each of these sequences we can obtain unbiased statistics from $p(u_t \mid u_{1:t-1}y_{1:t})$ using importance sampling [4]. We generate a set of independent samples $\{u^{(i,j)}_t : j = 1 \cdots s^{(i)}_t\}$ from a Gaussian distribution centered at $\hat{u}_t(u^{(i)}_{1:t-1})$ with variance proportional to $\hat{\sigma}_t(u^{(i)}_{1:t-1})$ and assign each sample a weight proportional to the ratio between the sampling distribution and the posterior distribution:

$$\hat{p}(u_t \mid u^{(i)}_{1:t-1}y_{1:t}) = \sum_{j=1}^{s^{(i)}_t} \delta(u_t - u^{(i,j)}_t) \frac{w_t(i,j)}{\sum_{k=1}^{s^{(i)}_t} w_t(i,k)} \tag{23}$$

$$w_t(i,j) = \frac{p(u^{(i,j)}_t \mid u^{(i)}_{t-1}) p(y_t \mid u^{(i)}_{1:t-1}u^{(i,j)}_t y_{1:t-1})}{g(u^{(i,j)}_t \mid \hat{u}_t(u^{(i)}_{1:t-1}), \alpha\hat{\sigma}_t(u^{(i)}_{1:t-1}))} \tag{24}$$

where $\hat{p}$ stands for an unbiased estimate of the corresponding probability term and $\alpha > 0$ is a parameter that determines the sharpness of the sampling distribution. As $\alpha \to 0$ we simply choose $\hat{u}_t(u_{1:t-1})$, the state that maximizes the posterior probability $p(u_t \mid u_{1:t-1}y_{1:t})$.

3.2 Credibility Equations

The credibility of the expert $u^{(i)}_{1:t-1}$ is proportional to the product of a prior term and a likelihood term

$$p(u^{(i)}_{1:t-1} \mid y_{1:t}) = \frac{p(u^{(i)}_{1:t-1} \mid y_{1:t-1}) p(y_t \mid u^{(i)}_{1:t-1}y_{1:t-1})}{p(y_t \mid y_{1:t-1})} \tag{25}$$

In Section 3.3 we explain how to obtain running estimates for the prior $p(u^{(i)}_{1:t-1} \mid y_{1:t-1})$. Regarding the likelihood, note that

$$\begin{aligned}p(y_t \mid u_{1:t-1}y_{1:t-1}) &= \int p(y_t u_t \mid u_{1:t-1}y_{1:t-1}) du_t \\ &= \int p(y_t \mid u_{1:t}y_{1:t-1}) p(u_t \mid u_{t-1}) du_t\end{aligned} \tag{26}$$

We already generated a set of samples $\{u^{(i,j)}_t : j = 1 \cdots s^{(i)}_t\}$ from $p(u_t \mid u^{(i)}_{1:t-1}y_{1:t})$. We can now use these samples to obtain an unbiased estimate of the likelihood

$$\begin{aligned}p(y_t \mid u^{(i)}_{1:t-1}y_{1:t-1}) &= \int p(y_t \mid u^{(i)}_{1:t-1}u_t y_{1:t-1}) p(u_t \mid u^{(i)}_{t-1}) du_t \\ &= \int p(y_t \mid u^{(i)}_{1:t-1}u_t y_{1:t-1}) g(u_t \mid \hat{u}_t(u^{(i)}_{1:t-1}), \hat{\sigma}_t(u^{(i)}_{1:t-1})) \\ &\qquad \frac{p(u_t \mid u^{(i)}_{t-1})}{g(u_t \mid \hat{u}_t(u^{(i)}_{1:t-1}), \hat{\sigma}_t(u^{(i)}_{1:t-1})} du_t \approx \frac{\sum_{j=1}^{s^{(i)}_t} w_t(i,j)}{s^{(i)}_t}\end{aligned} \tag{27}$$

If we only sample the most probable state $\hat{u}_t(u_{1:t-1})$ then the likelihood is approximated by the maximum value of the integrand.

3.3 Combining Opinion and Credibility

Opinion and credibility can be combined to obtain running estimates of the filtering distribution.

Initialization:

- Obtain n_1 samples $\{u_1^{(i)} : i = 1 \cdots n_1\}$ from $p(u_1)$. We refer to these samples as experts. For each expert the initial Gaussian prior distributions $p(v_1 b_1 \mid u_1^{(i)}) = p(v_1 b_1)$ are given as part of the model specification. The relative weight of the i^{th} expert, $r_1^{(i)}$ is set proportional to the probability of the image given the expert

$$r_1^{(i)} \propto p(y_1 \mid u_1^{(i)}) \tag{28}$$

and the weights are normalized to add up to one. This provides a Monte-Carlo estimate of the filtering distribution at the first time step:

$$\hat{p}(u_1 \mid y_1) = \sum_{i=1}^{n_1} r_{t-1}^{(i)} \delta(u_1 - u_1^{(i)}) \tag{29}$$

Update:

- By time $t-1$ we are given n_{t-1} pose experts $\{u_{i:t-1}^{(i)} : i = 1 \cdots n_{t-1}\}$. Each expert $u_{1:t-1}^{(i)}$ comes with a relative weight $r_t^{(i)}$ and with the mean and variance of the filtering distribution for texture given that expert, i.e., $E(V_{t-1}B_{t-1} \mid u_{1:t-1}^{(i)}, y_{1:t-1})$, $Var(V_{t-1}B_{t-1} \mid u_{1:t-1}^{(i)}, y_{1:t-1})$. The weights provide an estimate of the filtering distribution for pose at time $t-1$, which serves as the prior for time t

$$\hat{p}(u_{1:t-1} | y_{1:t-1}) = \sum_{i=1}^{n_{t-1}} r_{t-1}^{(i)} \delta(u_{1:t-1} - u_{1:t-1}^{(i)}) \tag{30}$$

For each expert, we compute the most probable pose $\hat{u}_t(u_{1:t1}^{(i)})$ and estimate the uncertainty about that pose $\hat{\sigma}_t(u_{1:t-1}^{(i)})$.

Based on the distribution of relative weights $\{r_{t-1}^{(i)} : i = 1 \cdots n_{t-1}\}$ we assign a number of descendants to each expert. This is usually known as a resampling step in the particle filtering literature [4], which discusses the pros and cons of different resampling rules. Suppose the resampling rule assigns $s_t^{(i)}$ descendants to expert i. We then generate as many independent samples $\{u_t^{(i,j)} : j = 1 \cdots s_t^{(i)}\}$ from the distribution $g(\cdot \mid \hat{u}_t(u_{1:t-1}^{(i)}), \hat{\sigma}_t(u_{1:t-1}^{(i)}))$, and compute the importance weight of each sample $w_t(i,j)$. This provides an estimate for the opinion

$$\hat{p}(u_t \mid u_{1:t-1}^{(i)} y_{1:t}) = \sum_{j=1}^{s_t^{(i)}} \delta(u_t - u_t^{(i,j)}) \frac{w_t(i,j)}{\sum_{k=1}^{s_t^{(i)}} w_t(i,k)} \tag{31}$$

and for the likelihood of each expert

$$\hat{p}(y_t \mid u_{1:t-1}^{(i)}) = \sum_{j=1}^{s_t^{(i)}} \frac{w_t(i,j)}{s_t^{(i)}} \tag{32}$$

The likelihood times the prior gives us the credibility of each expert

$$\hat{p}(u_{1:t-1}^{(i)} \mid y_{1:t}) \propto \frac{r_{t-1}^{(i)}}{s_t^{(i)}} \sum_{j=1}^{s_t^{(i)}} w_t(i,j) \tag{33}$$

From this we obtain $\hat{p}(u_{1:t} \mid y_{1:t})$,

$$\begin{aligned}
\hat{p}(u_{1:t} \mid y_{1:t}) &= \int \hat{p}(u_{1:t-1} \mid y_{1:t}) \\
&\quad \hat{p}(u_t \mid u_{1:t-1} y_{1:t}) du_{1:t-1} \qquad (34) \\
&= \sum_{i=1}^{n_{t-1}} \frac{\frac{r_{t-1}^{(i)}}{s_t^{(i)}} \sum_{j=1}^{s_t^{(i)}} w_t(i,j)}{\sum_{k=1}^{n_{t-1}} \frac{r_{t-1}^{(k)}}{s_t^{(k)}} \sum_{l=1}^{s_t^{(k)}} w_t(k,l)} \delta(u_{1:t-1} - u_{1:t-1}^{(i)}) \\
&\quad \sum_{m=1}^{s_t^{(i)}} \delta(u_t - u_t^{(i,m)}) \frac{w_t(i,m)}{\sum_{n=1}^{s_t^{(i)}} w_t(i,n)} \\
&= \sum_{i=1}^{n_{t-1}} \sum_{j=1}^{s_t^{(i)}} \delta(u_{1:t} - u_{1:t}^{(i)} u_t^{(i,j)}) \frac{\frac{r_{t-1}^{(i)}}{s_t^{(i)}} w_t(i,i)}{\sum_{k=1}^{n_{t-1}} \sum_{l=1}^{s_t^{(k)}} \frac{r_{t-1}^{(k)}}{s_{t-1}^{(k)}} w_t(k,l)} \qquad (35)
\end{aligned}$$

Note this behaves a set of experts $\{u_{1:t}^{(i)} : i = 1 \cdots n_t\}$ obtained by concatenating descendants to all the experts that generated them and dropping all the experts that did not generate any. The relative weight of the new expert $u_t^{(k)}$ formed by concatenating $u_t^{(i,j)}$ to $u_{1:t}^{(i)}$ is as follows

$$r_t^{(k)} \propto \frac{r_{t-1}^{(i)}}{s_t^{(i)}} w_t(i,j) \tag{36}$$

normalized so that the weights add up to one. In addition the texture opinions for each expert are found using the Kalman filter equations.

4 Tracking 3D deformable objects

The spatial location of the n points on the object varies with time due to rigid transformations (rotation, scale, translation) and non-rigid transformations (e.g., changes in expression). The rigid transformations are controlled by a rotation matrix R_t and a displacement vector D_t. The non-rigid transformations are modeled as linear combinations of a set of k 3-Dimensional

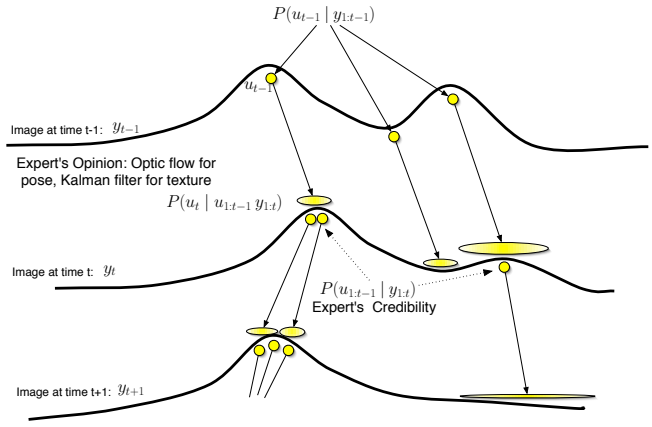

Figure 3: *An algorithm for solving the G-flow inference problem.*

morph keys $\phi(1), \cdots, \phi(k)$. Here $\phi(i)$ is an $n \times 3$ dimensional matrix, containing the 3-D position of the n object points on key-morph i. Let

$$\phi = (\phi(1), \cdots, \phi(k)) \tag{37}$$

be an $n \times 3k$ matrix containing all the morph keys. The vector $C_t \in \mathbb{R}^k$ contains the morph coefficients of the object at time t.

Let $X_t \in R^{2n}$ contain the 2-D coordinates of the projection of the n object points onto an image plane, i.e. X_{2i-1}, X_{2i} are the horizontal and vertical coordinates of the projection of the i^{th} point. Under weak perspective projection we have that

$$X_t = \beta U_t \tag{38}$$

where

$$\beta = \begin{pmatrix} 1 & b_{11}(1) & b_{12}(1) & \cdots & & & \\ 0 & 0 & \cdots & \cdots & & & \\ \vdots & \vdots & \vdots & \vdots & & & \\ 1 & b_{11}(n) & b_{12}(n) & \cdots & & & \\ 0 & 0 & \cdots & \cdots & & & \\ b_{3k}(1) & 0 & 0 & \cdots & \cdots & 0 & \\ 0 & 1 & b_{11}(1) & b_{12}(1) & \cdots & b_{3k}(1) & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ b_{3k}(n) & 0 & 0 & \cdots & \cdots & 0 & \\ 0 & 1 & b_{11}(n) & b_{12}(n) & \cdots & b_{3k}(n) & \end{pmatrix} \tag{39}$$

and

$$U_t = (D_t(1), R_t(1,1)C_t(1) \cdots R_t(1,3)C_t(k), D_t(2), R_t(2,1)C_t(1) \cdots, R_t(2,3)C_t(k))' \tag{40}$$

The matrix β contains the set of fixed animation morphs and the random variable U_t contains 3D pose and expression parameters. Standard techniques exist to recover the values of R_t and D_t once U_t is known [2].

To apply G-flow to this problem we need to find methods to find values for u_t that maximize $p(u_t \mid u_{1:t-1} y_{1:t})$.

Let $\bar{y}_t$ represent a matrix version of y_t, and $\bar{v}_t$ a matrix version of the object texture map $\bar{E}(\bar{V}_t \mid u_{1:t-1} y_{1:t-1})$. For the case in which the background is a white noise process, maximizing $p(u_t \mid u_{1:t-1} y_{1:t})$ is equivalent to minimizing

$$L(u_t, u_{t-1}) = \sum_{i=1}^{n} (\bar{y}_t(x_i) - \bar{v}_t(\beta u_t))^2 \tag{41}$$

Brand [2] showed that functions of this type can be optimized in real time using the Newton-Gauss algorithm.

5 Comparison to Other Approaches

Inference in G-flow belongs to a class of non-linear filtering problems known as "conditionally Gaussian problems". They can be solved using non-linear filtering techniques for the non-linear part and then propagate the solution to the linear part using time dependent Kalman filters. In the particle filtering literature this approach is known as Rao-Blackwelization [1].

A major problem with applications of particle filters to video tracking is the so called "Needle in a Haystack" problem (see Figure 4). The simplest approach to particle filtering starts with a set of samples from the filtering distribution at time $t-1$. For each particle samples are taken from the state transition distribution. Then the image at time t is observed and each particle is sample is weighted by the image likelihood function. Unfortunately in most tracking problem the likelihood function is highly peaked at the correct location (the needle) and relatively flat at the incorrect locations (the haystack). If the samples miss the peak of the likelihood function they will provide a very inefficient estimate of the filtering distribution. The problem gets worse as the number of parameters increases. For example, in 2D the likelihood function may not only be highly peaked but it may also have a strong orientation . Samples will be wasted by random sampling at the wrong location or with the wrong orientation (see Figure 5). Here we reduce this problem by explicitly computing the peak and orientation (i.e., precision matrix) of the opinion distribution distribution once the image has been observed. This is possible due to the the fact that the observed sequence (video images) is smooth in space and time, something that may not be necessarily the case for filtering problems in general.

[3] used an extended Kalman filter approach for a problem in which the 2D pose of an object and the texture of the object were tracked simultaneously. This limited the approach to unimodal solutions, which are known to be risky for tracking problems. They did not take advantage of the conditionally Gaussian nature

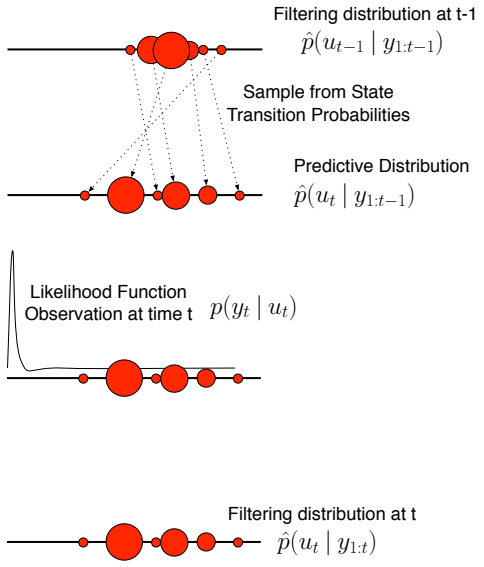

Figure 4: *A 1D version of the Needle in a Haystack Problem: If the likelihood function for the image at time t is very peaked, blind sampling approaches are likely to miss it and provide inefficient estimates of the filtering distribution. In G-flow this problem is reduced by explicitly computing the peak of the distribution after the image has been observed and sampling about that peak.*

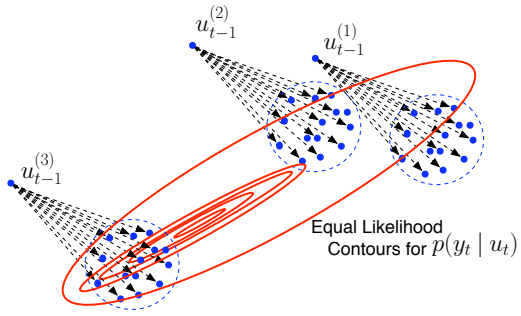

Figure 5: *A 2D version of the Needle in a Haystack Problem: The likelihood function is peaked and oriented. The descendants of particles $u_{t-1}^{(1)}$ and $u_{t-1}^{(2)}$ distribute about low likelihood regions due to poor location and blind sampling. The descendants of particle $u_{t-1}^{(3)}$ are well located but the sampling distribution does not have the right orientation, thus wasting a large number of particles. In G-flow the problem is reduced by explicitly computing the peak and orientation (i.e., the precision matrix) of the opinion distribution.*

of the problem and did not incorporate background information.

Brand [2] showed that one can combine the outputs of optic-flow solutions computed independently at different image points, along with their uncertainty, to find the rigid motion and non-rigid deformation parameters that best fit those flow solutions. We found that the approach is formally equivalent to directly propagating the linear constraints without intermediate computation of optic flow, which is the approach we use in our simulations. [6] presented an approach to propagate general non-rigid motion constraints on top the standard optic flow algorithm. Both [2] and [6] rely on unimodal state distributions, and do not learn object or background texture maps.

6 Simulations

We collected video of a person while making a variety of facial expression on command. An additional motion capture session was used to create a 3D model of the face and a set of 3D animation morphs. We are currently working on a system that will automatically find face geometry parameters based on a large dataset of 3D faces.

Twenty particles were initialized using the first frontal pose and propagated using the G-flow algorithm. A video of the entire sequence is available at our Web site. Figures 6 and 7 shows the distribution of particles for 3D pose and animation coefficients as a function of time. Note how the system can maintain multimodal distributions when necessary.

The system can run at about $1/n$ real time in Matlab, where n is the number of particles.

7 Conclusions

We presented a generative model (G-flow) for video sequences. The model provides a useful framework for studying the problem of how to dynamically combine motion and appearance information in a principled manner. Current optic flow and template based algorithms emerge in this model as optimal inference processes under specific conditions. In more realistic conditions optimal inference consists of a dynamically weighted combination of motion and appearance based information.

References

[1] C. Andrieu and A. Doucet N. de Freitas. Rao-blackwellised particle filtering via data augmentation. In *Advances in Neural Information Processing Systems*, number 13. MIT Press, Cambridge, Massachusetts, 2001.

[2] M. Brand. Flexible flow for 3D nonrigid tracking and shape recovery. In *CVPR*, 2001.

[3] F. Dellaert, S. Thrun, and C. Thorpe. Jacobian images of super-resolved texture maps for model-based motion estimation and tracking. In *Proc. IEEE Workshop Applications of Computer Vision*, 1998.

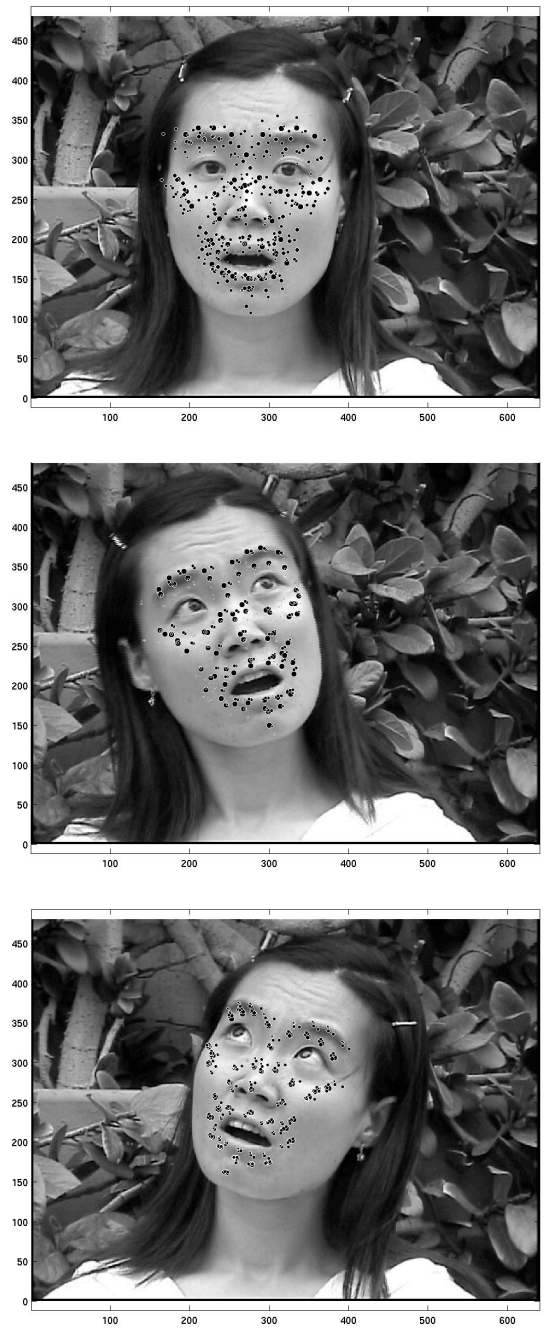

Figure 6: *This figure displays the locations of object points for 10 particles from an early frame (top) to a later frame (bottom) of the video sequence. The radius of the circles is proportional to the weight of the particles.*

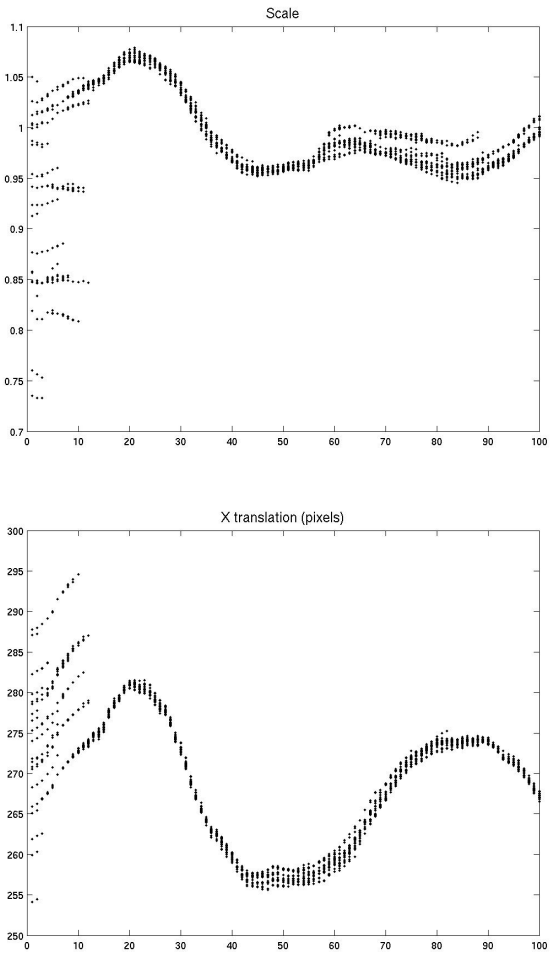

Figure 7: *Tracking results on a video sequence. There were 6 pose parameters and 4 morph parameters. The graphs contain the filtering distribution for 2 pose parameters using 20 particles. The continuous line connects the maximum posterior estimates.*

[4] George S. Fishman. *Monte Carlo Sampling: Concepts Algorithms and Applications.* Sprienger-Verlag, New York, 1996.

[5] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 1981.

[6] L. Torresani, D. Yang, G. Alexander, and C. Bregler. Tracking and modeling non-rigid objects with rank constraints. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2001.