Real-time 3-D Motion and Structure of Point Features: a Front-end System for Vision-based Control and Interaction

Hailin Jin Paolo Favaro Stefano Soatto

Washington University, One Brookings Dr. 1127, St. Louis - MO 63130
{hljin,fava,soatto}@essrl.wustl.edu

Abstract

We present a system that consists of one camera connected to a Personal Computer that can (a) select and track *a number of high-contrast point features on a sequence of images, (b)* estimate their three-dimensional motion and position *relative to an inertial reference frame, assuming rigidity, (c)* handle occlusions *that cause point-features to disappear as well as new features to appear. The system can also (d) perform partial self-calibration and (e) check for consistency of the rigidity assumption, although these features are not implemented in the current release. All of this is done* automatically and in real-time *(30Hz) for 40-50 point-features using commercial off-the-shelf hardware.*

The system is based on an algorithm presented by Chiuso et al. [3] whose properties have been analyzed in [2]. In particular, the algorithm is provably observable, provably minimal and provably stable - under suitable conditions. The core of the system, consisting of C++ code ready to interface with a frame grabber as well as Matlab code for development, is available at `http://ee.wustl.edu/~soatto/research.html`.

We demonstrate the system by showing its use as (1) an ego-motion estimator, (2) an object tracker, (3) an interactive input device. All without any modification of the system settings.

1 Introduction

We are interested in using vision as a sensor for machines to interact with the environment by moving, tracking, manipulating objects etc. In order to do so, a machine must be able to estimate its three-dimensional (3-D) motion relative to the scene and – to an extent that depends upon the application – the shape of the scene. The requirement of real-time operation imposes constraints on the algorithm to be used (it needs to be *causal*, for we cannot measure the future) as well as on the representation of shape (via point features; fancier representations cannot be handled by current commercial hardware). We consider, therefore, an N-tuple of points in the three-dimensional Euclidean space, represented by their image on a reference plane, $\mathbf{y}_0$ and their depth along the projection rays ρ. Let the points move under the action of a rigid motion represented by a translation vector T and a rotation matrix R, with linear velocity V and rotational velocity $\widehat{\omega}$, represented by a skew-symmetric 3×3 matrix using the "hat" notation (see [2] for details). We assume that we can measure the (noisy) projection $\mathbf{y}^i(t) = \pi\left(R(t)\mathbf{y}_0^i\rho^i + T(t)\right) + \mathbf{n}^i(t) \in \mathbb{R}^2 \;\; \forall\, i = 1 \ldots N$ which corresponds to an ideal perspective projection model. This choice is not crucial and the discussion can be easily extended to other projection models (e.g. spherical, orthographic, para-perspective, etc.). By organizing the time-evolution of the configuration of points and their motion, we end up with a discrete-time, non-linear dynamical system. In [2] it is proven that the following model is minimal, and that the Extended Kalman Filter (EKF) based on it is stable (in mean-square and with probability one):

$$\begin{cases} \mathbf{y}_0^i(t+1) = \mathbf{y}_0^i(t) & i = 4 \ldots N & \mathbf{y}_0^i(0) = \mathbf{y}_0^i \\ \rho^i(t+1) = \rho^i(t) & i = 2 \ldots N & \rho^i(0) = \rho_0^i \\ T(t+1) = \exp(\widehat{\omega}(t))T(t) + V(t) & & T(0) = T_0 \\ \Omega(t+1) = Log_{SO(3)}(\exp(\widehat{\omega}(t))\exp(\widehat{\Omega}(t))) & & \Omega(0) = \Omega_0 \\ V(t+1) = V(t) + \alpha_V(t) & & V(0) = V_0 \\ \omega(t+1) = \omega(t) + \alpha_\omega(t) & & \omega(0) = \omega_0 \\ \mathbf{y}^i(t) = \pi\left(\exp(\widehat{\Omega}(t))\mathbf{y}_0^i(t)\rho^i(t) + T(t)\right) + \mathbf{n}^i(t) & & \\ \qquad \text{where} \quad \mathbf{n}^i \sim \mathcal{N}(0, \Sigma_n) & i = 1 \ldots N. & \end{cases} \tag{1}$$

The notation $Log_{SO(3)}(R)$ stands for Ω such that $R = e^{\widehat{\Omega}}$ and is computed by inverting Rodrigues' formula[1]. Note that the indices of $\mathbf{y}_0$ and ρ in the state start from 4 and 2 respectively. This is to guarantee that the model is minimal and stable[2].

2 Implementation

The implementation of the EKF based on the above model is discussed in detail in [3], including partial self-calibration and handling of occlusions. Here we only describe the

[1] A Matlab implementation of $Log_{SO(3)}$ is included in the software distribution.

[2] There are in the literature similar models that do not include $\mathbf{y}_0$ in the state, for instance [1]. It is shown in [2] that such models are submimimal, and therefore have a non-zero-mean measurement error that results in a highly biased estimate.

"recipe" algorithm in a level of detail that should be sufficient to follow the implementation made available on the Web, or to implement the algorithm independently.

Initialization Choose the initial conditions $\mathbf{y}_0^i = \mathbf{y}^i(0)$; $\rho_0^i = 1$; $T_0 = 0$; $\Omega_0 = 0$; $V_0 = 0$; $\omega_0 = 0$ for $i = 1 \dots N$ where N is chosen depending upon the performance of the hardware ($N = 50$ in our case, a dual PIII 733Mhz system). For the initial variance P_0, choose it to be block diagonal with blocks $\Sigma_{n^i}(0)$ given from the analysis of the feature-tracking algorithm[3] corresponding to $\mathbf{y}_0^i$, a large positive number M (typically 100-1000 units of focal length) corresponding to ρ^i, zeros corresponding to T_0 and Ω_0 (fixing the inertial frame to coincide with the initial reference frame). We also choose a large positive number W for the blocks corresponding to V_0 and ω_0. The variance $\Sigma_w(t)$ is a design parameter that is available for tuning. Finally, set $P(0|0) = P_0$ and $\hat{\xi}(0|0) \doteq [\mathbf{y}^4{}_0^T, \dots, \mathbf{y}^N{}_0^T, \rho_0^2, \dots, \rho_0^N, T_0^T, \Omega_0^T, V_0^T, \omega_0^T]^T$

Transient During the first transient of the filter, we do not allow for new features to be acquired. Whenever a feature is lost, its state is removed from the model and its best current estimate is placed in a storage vector. The transient can be tested, for instance, by a threshold on the innovation, a threshold on the variance of the estimates, or by a fixed time interval. We choose a combination of them, with the time interval set to 30 frames, corresponding to the first second of video. The recursion to update the state ξ and the variance P proceed as follows: Let f and h denote the state and measurement model, so that equation (1) can be written in concise form as

$$\begin{cases} \xi(t+1) = f(\xi(t)) + w(t) & w(t) \sim \mathcal{N}(0, \Sigma_w) \\ y(t) = h(\xi(t)) + n(t) & n(t) \sim \mathcal{N}(0, \Sigma_n) \end{cases} \quad (2)$$

Prediction:

$$\begin{cases} \hat{\xi}(t+1|t) = f(\hat{\xi}(t|t)) \\ P(t+1|t) = F(t)P(t|t)F^T(t) + \Sigma_w \end{cases}$$

Update:

$$\begin{cases} \hat{\xi}(t+1|t+1) = \hat{\xi}(t+1|t) + L(t+1)\left(y(t+1) - h(\hat{\xi}(t+1|t))\right) \\ P(t+1|t+1) = \Gamma(t+1)P(t+1|t)\Gamma^T(t+1) + \\ \qquad + L(t+1)\Sigma_n(t+1)L^T(t+1). \end{cases}$$

Gain:

$$\begin{cases} \Gamma(t+1) \doteq I - L(t+1)C(t+1) \\ L(t+1) \doteq P(t+1|t)C^T(t+1)\Lambda^{-1}(t+1) \\ \Lambda(t+1) \doteq C(t+1)P(t+1|t)C^T(t+1) + \Sigma_n(t+1) \end{cases}$$

Linearization:

$$F(t) \doteq \begin{bmatrix} I_{3N} & 0 & 0 & 0 \\ & \exp(\widehat{\omega}) & 0 & I_3 & \frac{\partial (e^{\widehat{\omega}} T)}{\partial \omega} \\ & \frac{\partial Log_{SO(3)}(e^{\widehat{\omega}} e^{\widehat{\Omega}})}{\partial \Omega} & 0 & \frac{\partial Log_{SO(3)}(e^{\widehat{\omega}} e^{\widehat{\Omega}})}{\partial \omega} & 0 \\ & & I_3 & & I_3 \end{bmatrix}$$

$$C(t) = \Pi \cdot \begin{bmatrix} \hat{Y}^1 & 0 & \dots & \hat{R}^1 & 0 & \dots & I_3 & \frac{\partial \hat{\Theta}^1}{\partial \Omega} & 0 & 0 \\ & \ddots & & & \ddots & & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \hat{Y}^N & 0 & \dots & \hat{R}^N & I_3 & \frac{\partial \hat{\Theta}^N}{\partial \Omega} & 0 & 0 \end{bmatrix}$$

where we set $\hat{Y}^i = e^{\widehat{\Omega}} \begin{bmatrix} I_2 \\ \mathbf{0} \end{bmatrix} \rho_i$, $\hat{R}^i = e^{\widehat{\Omega}} \begin{bmatrix} \mathbf{y}_0^i \\ 0 \end{bmatrix}$ and $\hat{\Theta}^i = e^{\widehat{\Omega}} \begin{bmatrix} \mathbf{y}_0^i \\ 1 \end{bmatrix} \rho_i$.

The matrix Π is block-diagonal with blocks $\frac{1}{\rho^i} \begin{bmatrix} I_2 & -\hat{y}^i \end{bmatrix}$ where $\hat{y}^i = \pi\left(e^{\Omega} \mathbf{y}_0^i \rho^i + T\right)$.

Regime Whenever a feature disappears, we remove it from the state as during the transient. However, during regime operation a feature selection module works in parallel with the filter to select new features so as to maintain roughly a constant number N and a distribution as uniform as possible across the image plane. We implement this by randomly sampling points on the plane, then searching around that location for a feature that passes a "sum of square difference"-type test. Once a new point-feature is found, a "subfilter" is initialized. Its evolution is given by

$$\textbf{Initialization:} \begin{cases} \hat{\mathbf{y}}_\tau^i(\tau|\tau) = \mathbf{y}_\tau^i(\tau) \\ \hat{\rho}_\tau^i(\tau|\tau) = 1 \\ P_\tau^i(\tau|\tau) = \begin{bmatrix} \Sigma_{n^i}(\tau) & \\ & M \end{bmatrix} \end{cases}$$

$$\textbf{Prediction:} \begin{cases} \hat{\mathbf{y}}_\tau^i(t+1|t) = \hat{\mathbf{y}}_\tau^i(t|t) \\ \hat{\rho}_\tau^i(t+1|t) = \hat{\rho}_\tau^i(t|t) \\ P_\tau(t+1|t) = P_\tau(t+1|t) + \Sigma_w(t) \end{cases} \quad t > \tau$$

$$\textbf{Update:} \begin{bmatrix} \hat{\mathbf{y}}_\tau^i(t+1|t+1) \\ \hat{\rho}_\tau^i(t+1|t+1) \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{y}}_\tau^i(t+1|t) \\ \hat{\rho}_\tau^i(t+1|t) \end{bmatrix} + L_\tau(t+1)\left(\mathbf{y}^i(t) - \pi(\exp(\widehat{\Omega}(t)) \left[\exp(\widehat{\Omega}(\tau))\right]^{-1} \left[\mathbf{y}^i(t)\rho^i(t) - T(\tau)\right] + T(t))\right)$$

and P_τ is updated according to the usual Riccati equation. After a probation period the feature is inserted into the state.

Tuning The variance $\Sigma_w(t)$ is a design parameter which we choose to be block diagonal with the blocks corresponding to $T(t)$ and $\Omega(t)$ equal to zero (a deterministic integrator). We choose the remaining parameters using the Periodogram test. In practice, we choose the blocks corresponding to $\mathbf{y}_0^i$ equal to the variance of the measurements, and the elements corresponding to ρ^i all equal to σ_ρ. We then choose the blocks corresponding to V and ω to be diagonal with element σ_v, and then we change σ_v relative to σ_ρ depending on whether we want to allow for more or less regular motions. We then change both, relative to the variance of the measurement noise, depending on the level of desired smoothness in the estimates.

Acknowledgments

Supported by NSF grant IIS-9876145 and ARO grant DAAD19-99-1-0139. The authors wish to thank Xiaolin Feng and Pietro Perona for their generous sharing of their code for feature tracking.

[3] We assume that the tracking error is independent in each point, and therefore Σ_n is block diagonal with diagonal equal to 1 pixel std. in the current implementation.

References

[1] A. Azarbayejani and A. Pentland. Recursive estimation of motion, structure and focal length. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(6):562–575, 1995.

[2] A. Chiuso and S. Soatto. 3-D Motion and structure from 2-D motion causally integrated over time. Part I: theory. Tutorial lecture notes. IEEE Conf. on Robotics and Automation, April 2000.

[3] A. Chiuso and P. Favaro and H. Jin and S. Soatto. 3-D motion and structure causally integrated over time. Part 2: experiments. In Proc. of the Eur. Conf. on Computer Vision, June 2000 (in press).