general feature-tracking context.

In Algorithm 4.1 we summarize a basic algorithm for feature tracking or optical flow; a more effective version of this algorithm that involves a multi-resolution representation and subpixel refinement is described in Chapter 11 (Algorithm 11.2).

Algorithm 4.1 (Basic feature tracking and optical flow).

Given an image $I(\boldsymbol{x})$ at time t, set a window W of fixed size, use the filters given in Appendix 4.A to compute the image gradient (I_x, I_y), and compute $G(\boldsymbol{x}) \doteq \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$ at every pixel $\boldsymbol{x}$. Then, either

- (feature tracking) select a number of point features by choosing $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots$ such that $G(\boldsymbol{x}_i)$ is invertible, or
- (optical flow) select $\boldsymbol{x}_i$ to be on a fixed grid.

An invertibility test of G that is more robust to the effects of noise will be described in Algorithm 4.2.

- Compute $\boldsymbol{b}(\boldsymbol{x}, t) \doteq \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$.
- If $G(\boldsymbol{x})$ is invertible (which is guaranteed for point features), compute the displacement $\boldsymbol{u}(\boldsymbol{x}, t)$ from equation (4.20). If $G(\boldsymbol{x})$ is not invertible, return $\boldsymbol{u}(\boldsymbol{x}, t) = 0$.

The displacement of the pixel $\boldsymbol{x}$ at time t is therefore given by $\boldsymbol{u}(\boldsymbol{x}, t) = -G(\boldsymbol{x})^{-1}\boldsymbol{b}(\boldsymbol{x}, t)$ wherever $G(\boldsymbol{x})$ is invertible.

- (Feature tracking) at time $t + 1$, repeat the operation at $\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x}, t)$.
- (Optical flow) at time $t + 1$, repeat the operation at $\boldsymbol{x}$.

4.3.2 Large baseline: affine model and normalized cross-correlation

The small-baseline tracking algorithm presented in the previous section results in very efficient and fast implementations. However, when features are tracked over an extended time period, the estimation error resulting from matching templates between two adjacent frames accumulates in time. This leads to eventually losing track of the originally selected features. To avoid this problem, instead of matching image regions between adjacent frames, one could match image regions between the initial frame, say I_1, and the current frame, say I_2. On the other hand, the deformation of the image regions between the first frame and the current frame can no longer be modeled by a simple translational model. Instead, a commonly adopted model is that of affine deformation of image regions that support point