

experimental comparison of some of these algorithms on image retrieval datasets. You can also find more details on related techniques and systems in Section 6.2.3 on visual similarity search, which discusses global descriptors that represent an image with a single vector (Anandjelovic, Gronat et al. 2016; Radenović, Tolias, and Chum 2019; Yang, Kien Nguyen et al. 2019; Cao, Araújo, and Sim 2020; Ng, Balntas et al. 2020; Tolias, Jenicock, and Chum 2020) as alternatives to bags of local features, Section 11.2.3 on location recognition, and Section 11.4.6 on large-scale 3D reconstruction from community (*internet*) photos.

### 7.1.5 Feature tracking

An alternative to independently finding features in all candidate images and then matching them is to find a set of likely feature locations in a first image and to then search for their corresponding locations in subsequent images. This *detect then track* approach is more widely used for video tracking applications, where the expected amount of motion and appearance deformation between adjacent frames is expected to be small.

The process of selecting good features to track is closely related to selecting good features for more general recognition applications. In practice, regions containing high gradients in both directions, i.e., which have high eigenvalues in the auto-correlation matrix (7.8), provide stable locations at which to find correspondences (Shi and Tomasi 1994).

In subsequent frames, searching for locations where the corresponding patch has low squared difference (7.1) often works well enough. However, if the images are undergoing brightness change, explicitly compensating for such variations (9.9) or using normalized cross-correlation (9.11) may be preferable. If the search range is large, it is also often more efficient to use a *hierarchical search strategy*, which uses matches in lower-resolution images to provide better initial guesses and hence speed up the search (Section 9.1.1). Alternatives to this strategy involve learning what the appearance of the patch being tracked should be and then searching for it in the vicinity of its predicted position (Avidan 2001; Jurie and Dhume 2002; Williams, Blake, and Cipolla 2003). These topics are all covered in more detail in Section 9.1.3.

If features are being tracked over longer image sequences, their appearance can undergo larger changes. You then have to decide whether to continue matching against the originally detected patch (feature) or to re-sample each subsequent frame at the matching location. The former strategy is prone to failure, as the original patch can undergo appearance changes such as foreshortening. The latter runs the risk of the feature drifting from its original location to some other location in the image (Shi and Tomasi 1994). (Mathematically, small misregistration errors compound to create a *Markov random walk*, which leads to larger drift over time.)

Jenicek, and Chum 2020) as alternatives to bags of local features, Section 11.2.3 on location recognition, and Section 11.4.6 on large-scale 3D reconstruction from community (internet) photos.

### 7.1.5 Feature tracking

An alternative to independently finding features in all candidate images and then matching them is to find a set of likely feature locations in a first image and to then *search* for their corresponding locations in subsequent images. This kind of *detect then track* approach is more widely used for video tracking applications, where the expected amount of motion and appearance deformation between adjacent frames is expected to be small.

The process of selecting good features to track is closely related to selecting good features for more general recognition applications. In practice, regions containing high gradients in both directions, i.e., which have high eigenvalues in the auto-correlation matrix (7.8), provide stable locations at which to find correspondences (Shi and Tomasi 1994).

In subsequent frames, searching for locations where the corresponding patch has low squared difference (7.1) often works well enough. However, if the images are undergoing brightness change, explicitly compensating for such variations (9.9) or using *normalized cross-correlation* (9.11) may be preferable. If the search range is large, it is also often more efficient to use a *hierarchical* search strategy, which uses matches in lower-resolution images to provide better initial guesses and hence speed up the search (Section 9.1.1). Alternatives to this strategy involve learning what the appearance of the patch being tracked should be and then searching for it in the vicinity of its predicted position (Avidan 2001; Jurie and Dhome 2002; Williams, Blake, and Cipolla 2003). These topics are all covered in more detail in Section 9.1.3.

If features are being tracked over longer image sequences, their appearance can undergo larger changes. You then have to decide whether to continue matching against the originally detected patch (feature) or to re-sample each subsequent frame at the matching location. The former strategy is prone to failure, as the original patch can undergo appearance changes such as foreshortening. The latter runs the risk of the feature drifting from its original location to some other location in the image (Shi and Tomasi 1994). (Mathematically, small misregistration errors compound to create a *Markov random walk*, which leads to larger drift over time.)

A preferable solution is to compare the original patch to later image locations using an *affine* motion model (Section 9.2). Shi and Tomasi (1994) first compare patches in neighboring frames using a translational model and then use the location estimates produced by this to initialize an affine registration between the patches in the next frame, and then repeat this process.

experimental comparison of some of these algorithms on image retrieval dataset. You can also find more details on related techniques and systems in Section 6.2.3 on visual similarity search, which discusses global descriptors that represent an image with a single vector (Andjeljević, Grauman et al. 2016; Radenović, Tolia, and Chua 2019; Yang, Keen, Narvekar et al. 2019; Cao, Aranjo, and Sim 2020; Ng, Barnes et al. 2020; Tolka, Jeniack, and Chua 2020) as alternatives to bags of local features. Section 11.2 on location recognition, and Section 11.4.6 on large-scale 3D reconstruction from community (internet) photos.

# 7.1.5 Feature tracking

An alternative to independently finding features in all candidate images and then matching them is to find a set of  $N$  feature locations in a single image and to then search for their corresponding locations in other images. This kind of matching is much faster, and it is more widely used for video tracking applications, where it is expected that the appearance of certain regions in successive frames will change little.

The process of electron cloud rotation is called molecular rotation. It is similar to the rotation of molecules in the liquid state, but it is much faster. The rotation of molecules in the liquid state is called molecular rotation. It is similar to the rotation of molecules in the liquid state, but it is much faster.

In subsequent frame searching for location, we can use the squared difference (7.1) or a weighted sum of squared differences in brightness change, capturing both motion and blur. The cross-correlation (9.11) may be used to measure the similarity between frames. This is an efficient approach for hierarchical search, as it provides both initial guess and confidence measure to this search, involving iterative refinement. The search can then search for translation only, constrained by the initial guess. Williams, Blakes, and Cohen (2002) and Williams and Shah (2003) have shown that this approach is very effective.

- If features are better tracked, we can ignore larger changes. You can have a detection patch (feature) as 10x10 pixels, so the former strategy is prone to failure. This is known as foreshortening. The image will be rotated at some other location in the image. If we do not take care of this, then we will get a registration error across consecutive frames (time).