

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 (Arandjelović, Gronat *et al.* 2016; Radenović, Tolias, and Chum 2019; Yang, Kien Nguyen *et al.* 2019; Cao, Araujo, and Sim 2020; Ng, Balntas *et al.* 2020; Tolias, Jenkoek, 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.)

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 (Aranjuez, Gout et al. 2006; Rasmovic, Todor, and Chua 2009; Tang, Kim, Nijssen et al. 2019; Cao, Arango, and Sim 2020; Ng, Balasari et al. 2020; Tolan, Kemel, and Chua 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.

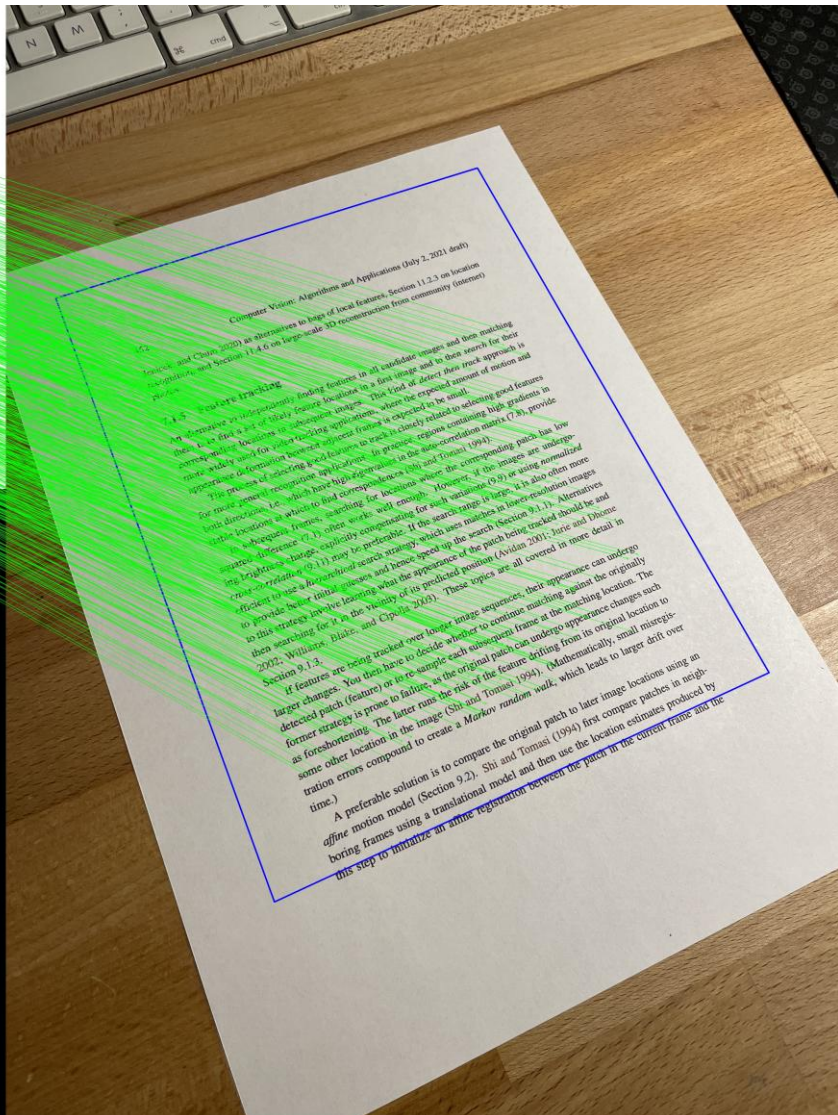
7.1.5 Feature tracking

An alternative to independently finding features in all candidate images and then matching them is to find a set of *affine* feature locations in a first image and to then search for their corresponding locations in subsequent images. This kind of *affine* (i.e., linear) 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 is often (though not necessarily) aided by selecting good frames for more general recognition applications, by detecting frames containing high gradients in both directions, i.e., which have high local values in the image gradient matrix ∇I at local stable locations at which to find corresponding locations in later frames.

In subsequent frames, subpixel-precise location matching is often done using a least-squares difference (LSD) cost function. However, if the image is undergoing high-speed motion, especially nonrigid motion, using a least-squares cost function is inefficient to use without some sort of search heuristic, which may reduce its power to find the best match. To provide better global guidance and hence speed up the search process, Shi and Tomasi (1994) proposed to this strategy with a heuristic (called *corner response*) for patch selection. They then searched for a good match in the vicinity of the predicted position (Avidan 2001; Joris and Bloesch 2002; Williams, Bialko, and Cipora 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 with the original patch, or to detect a new patch. In this case, you may want to use a more robust cost function, such as *robustness*, to provide a better patch selection. The latter can be done by either detecting a new patch, or by using other locations in the image (called *feature matching*) to find a better match. This is often done using a *Markov random walk* (MRF) model, which is a probabilistic model for feature matching over time.)



Here **SIFT** is used for feature matching.

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, [July 2, 2021 \(draft\)](#) (Arandjelovic, Gromat *et al.* 2016; Radenović, Tolias, and Chum 2019; Yang, Kien Nguyen *et al.* 2019; Cao, Araujo, and Sim 2020; Ng, Balntas *et al.* 2020; Teitiss/19/0/3 on location 2020) as alternatives to bags of local features, Section 11.4.3 on location, [community \(internet\)](#) 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.)

Now the original patch to later image locations using an Shi and Tomasi (1994) first compare patches in neighborhood and then use the location estimates produced by

