Hessian after Convergence: A New Perspective on Lucas Kanade Tracking

Abhineet Kumar Singh
Student ID: 1395723

1 SSD

Eq. 1 gives the objective function that is minimized by the SSD based Lucas Kanade (LK) tracker at time t using the forward compositional (FC) formulation:

$$F_{ssd}(\Delta\mathbf{p}_t) = \sum_{\mathbf{x}} \left[I_t(\mathbf{w}(\mathbf{w}(\mathbf{x}, \mathbf{p}_{t-1}), \Delta\mathbf{p}_t)) - I_0(\mathbf{x}) \right] \tag{1}$$

where I_t and I_0 are the current and initial images respectively, $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ holds the pixel coordinates for the initial object patch (or template) sampled with n pixels and $I(\mathbf{x})$ refers to the pixel values in image I at locations $\mathbf{x}$ flattened as a vector of size n. Further, $\mathbf{w}$ is the warp function (provided by the state space model), $\mathbf{w}(\mathbf{x}, \mathbf{p})$ refers to the locations of pixels $\mathbf{x}$ after being warped by $\mathbf{w}$ using parameters $\mathbf{p}$, $\mathbf{w}(\mathbf{x}, \mathbf{p}_{t-1})$ is the latest estimate of the object location (from frame I_{t-1}) and $\Delta\mathbf{p}_t$ is the incremental update to $\mathbf{p}_{t-1}$ that we are trying to estimate and that will be used to update it as:

$$\mathbf{p}_t = \mathbf{p}_{t-1} \circ \Delta\mathbf{p}_t \tag{2}$$

Using the Newton method, the incremental update to warp parameters is computed in closed form as:

$$\Delta\mathbf{p}_t = -H_{ssd}^{-1} G_{ssd}^T \tag{3}$$

where $H_{ssd} = \dfrac{\partial^2 F_{ssd}}{\partial \mathbf{p}^2}$ is the $S \times S$ Hessian matrix and $G_{ssd} = \dfrac{\partial F_{ssd}}{\partial \mathbf{p}}$ is the $1 \times S$ gradient vector of F_{ssd} w.r.t. SSM parameters $\mathbf{p} = \{p_1, p_2, ..., p_S\}$. The expression for the **Full Newton (FN)** Hessian is given in Eq. 4:

$$H_{ssd}^{fn} = \sum_{\mathbf{x}} \left\{ \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} \right]^T \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} \right] + (I_t - I_0) \frac{\partial^2 I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}^2} \right\} \tag{4}$$

with

$$\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} = \nabla I_t^{\mathbf{w}}(\mathbf{x}) \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \tag{5}$$

and

$$\frac{\partial^2 I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}^2} = \left[\frac{\partial \mathbf{w}}{\partial \mathbf{p}} \right]^T \nabla^2 I_t^{\mathbf{w}}(\mathbf{x}) \left[\frac{\partial \mathbf{w}}{\partial \mathbf{p}} \right] + \nabla I_t^{\mathbf{w}}(\mathbf{x}) \frac{\partial^2 \mathbf{w}}{\partial \mathbf{p}^2} \tag{6}$$

where $I_t^{\mathbf{w}}(\mathbf{x}) = I_t(\mathbf{w}(\mathbf{x}, \mathbf{p}_{t-1}))$ is the image I_t warped using the latest estimate of SSM parameters, $\nabla I(\mathbf{x})$ is the 1×2 gradient and $\nabla^2 I(\mathbf{x})$ is the 2×2 hessian of image I computed at

pixel location $\mathbf{x}$ and $\frac{\partial \mathbf{w}}{\partial \mathbf{p}}$ is the $2 \times S$ Jacobian of the warping function w.r.t. SSM parameters evaluated at their *initial* values ($\mathbf{p}_0$).

This expression, however, is rarely used in practice and instead it is the so called **Gauss Newton (GN)** Hessian that is usually used and also works best for SSD. This is given as:

$$\begin{aligned} H_{ssd}^{gn} &= \sum_{\mathbf{x}} \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}\right]^T \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}\right] \\ &= \left[\nabla I_t^{\mathbf{w}}(\mathbf{x}) \frac{\partial \mathbf{w}}{\partial \mathbf{p}}\right]^T \left[\nabla I_t^{\mathbf{w}}(\mathbf{x}) \frac{\partial \mathbf{w}}{\partial \mathbf{p}}\right] \end{aligned} \tag{7}$$

This expression is conventionally interpreted as a first order approximation of the expression in Eq. 4 where the second order term has been ignored as being too costly to compute and too small to matter especially close to convergence.

2 MI

Eq. 8 presents the objective function that is maximized by the MI based Lucas Kanade tracker:

$$F_{mi}(\Delta \mathbf{p}_t) = \sum_{i,j} P_{I_t I_0}(i,j) \log\left(\frac{P_{I_t I_0}(i,j)}{P_{I_t}(i) P_{I_0}(j)}\right) \tag{8}$$

where $P_{I_t I_0}$ is the normalized joint histogram between $I_t^{\mathbf{w}}(\mathbf{w}(\mathbf{x}, \Delta \mathbf{p}_t))$ and $I_0(\mathbf{x})$, P_{I_t} and P_{I_0} are the corresponding marginal histograms and the summation is carried out over all bins in the histograms.

The FN Hessian for MI is given as:

$$H_{mi}^{fn}(t) = \sum_{i,j} \left\{ \frac{\partial P_{I_t I_0}(i,j)}{\partial \mathbf{p}}^T \frac{\partial P_{I_t I_0}(i,j)}{\partial \mathbf{p}} \left(\frac{1}{P_{I_t I_0}(i,j)} - \frac{1}{P_{I_t}(i)}\right) + \frac{\partial^2 P_{I_t I_0}(i,j)}{\partial \mathbf{p}^2} \left(1 + \log\left(\frac{P_{I_t I_0}(i,j)}{P_{I_t}(i)}\right)\right) \right\} \tag{9}$$

with

$$\begin{aligned} \frac{\partial P_{I_t I_0}(i,j)}{\partial \mathbf{p}} &= \frac{1}{n} \sum_{\mathbf{x}} \frac{\partial \phi(i - I_t^{\mathbf{w}}(\mathbf{x}))}{\partial \mathbf{p}} \phi(j - I_0(\mathbf{x})) \\ &= \frac{1}{n} \sum_{\mathbf{x}} -\frac{\partial \phi}{\partial i} \frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} \phi(j - I_0(\mathbf{x})) \end{aligned} \tag{10}$$

and

$$\begin{aligned} \frac{\partial^2 P_{I_t I_0}(i,j)}{\partial \mathbf{p}^2} &= \frac{1}{n} \sum_{\mathbf{x}} \left\{ \frac{\partial^2 \phi(i - I_t^{\mathbf{w}}(\mathbf{x}))}{\partial \mathbf{p}^2} \phi(j - I_0(\mathbf{x})) \right\} \\ &= \frac{1}{n} \sum_{\mathbf{x}} \left\{ \left(\frac{\partial^2 \phi}{\partial i^2} \frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}^T \frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} - \frac{\partial \phi}{\partial i} \frac{\partial^2 I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}^2} \right) \phi(j - I_0(\mathbf{x})) \right\} \end{aligned} \tag{11}$$

where $\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}$ and $\frac{\partial^2 I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}^2}$ are as defined in Eqs. 5 and 6 respectively.

Now, using the conventional interpretation of Eq. 7 as the *first order approximation* of Eq. 4, the corresponding GN Hessian for MI will be:

$$H_{mi}^{gn}(t) = \sum_{i,j} \frac{\partial P_{I_t I_0}(i,j)}{\partial \mathbf{p}}^T \frac{\partial P_{I_t I_0}(i,j)}{\partial \mathbf{p}} \left(\frac{1}{P_{I_t I_0}(i,j)} - \frac{1}{P_{I_t}(i)}\right) \tag{12}$$

However, as mentioned in [1, 2] and confirmed by my own experiments, this Hessian does not work at all and causes the tracker to fail even before the object starts moving. Though the GN Hessian has been used before for MI based tracking [3] as well as for general image registration[4, 5], the results reported there do not agree with my experience nor with the results provided in [1, 2]. In fact if the Wikipedia article on GN [6] is correct, this method "unlike Newton's method, can only be used to minimize a sum of squared function values". In that case, it seems absurd to even expect it to work with MI as it has no concept of least squares at all.

In addition, even using the full Newton Hessian (Eq. 9) does not perform well enough to be useful for any practical tracking as it invariably fails soon after the object starts moving. Also, this behavior of Newton method is not unique to MI since SSD shows poorer performance when using the second order Hessian (Eq. 4) too. The fact that Newton method does not perform as well as the Gauss Newton method for SSD has also been observed before [7] through decreased frequency of convergence in static experiments. The reason suggested there was that the presence of noise in the image makes the numerically computed image Hessian $\frac{\partial^2 I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}^2}$ and thus the second order SSD Hessian H_{ssd}^{fn} noisy enough to outweigh the advantage of the extra accuracy. The reason proposed in [1, 2], however, is that the second order Hessian makes the domain/range of convergence too narrow due to which the algorithm fails to converge for even small inter frame motions. The solution mentioned there is to instead use an estimate of the second order **Hessian after convergence** (**HAC**), i.e. an estimate of the expression in Eq. 9 computed by replacing $I_t^{\mathbf{w}}(\mathbf{x})$ with $I_t^{\bar{\mathbf{w}}}(\mathbf{x}) = I_t(\mathbf{w}(\mathbf{w}(\mathbf{x}, \mathbf{p}_{t-1}), \Delta\bar{\mathbf{p}}_t))$ where $\Delta\bar{\mathbf{p}}_t = \arg\max_{\Delta\mathbf{p}} F_{mi}(\Delta\mathbf{p})$. As $\Delta\bar{\mathbf{p}}_t$ is not known, $I_t^{\bar{\mathbf{w}}}$ was approximated in [1, 2] by assuming perfect alignment after convergence, i.e. $I_t^{\bar{\mathbf{w}}}(\mathbf{x}) = I_0(\mathbf{x})$. Let the resultant estimate be known as the **Initial Newton (IN)** Hessian or H_{mi}^{in} and given as:

$$H_{mi}^{in} = \sum_{i,j} \left[\frac{\partial P_{I_0 I_0}(i,j)}{\partial \mathbf{p}}^T \frac{\partial P_{I_0 I_0}(i,j)}{\partial \mathbf{p}} \left(\frac{1}{P_{I_0 I_0}(i,j)} - \frac{1}{P_{I_0}(i)} \right) + \frac{\partial^2 P_{I_0 I_0}(i,j)}{\partial \mathbf{p}^2} \left(1 + \log \left(\frac{P_{I_0 I_0}(i,j)}{P_{I_0}(i)} \right) \right) \right] \tag{13}$$

with

$$\frac{\partial P_{I_0 I_0}(i,j)}{\partial \mathbf{p}} = \frac{1}{n} \sum_{\mathbf{x}} -\frac{\partial \phi}{\partial i} \frac{\partial I_0(\mathbf{x})}{\partial \mathbf{p}} \phi(j - I_0(\mathbf{x})) \tag{14}$$

and

$$\frac{\partial^2 P_{I_0 I_0}(i,j)}{\partial \mathbf{p}^2} = \frac{1}{n} \sum_{\mathbf{x}} \left\{ \left(\frac{\partial^2 \phi}{\partial i^2} \frac{\partial I_0(\mathbf{x})}{\partial \mathbf{p}}^T \frac{\partial I_0(\mathbf{x})}{\partial \mathbf{p}} - \frac{\partial \phi}{\partial i} \frac{\partial^2 I_0(\mathbf{x})}{\partial \mathbf{p}^2} \right) \phi(j - I_0(\mathbf{x})) \right\} \tag{15}$$

In addition to working much better than H_{mi}^{fn} and H_{mi}^{gn}, this has the additional advantage that, being independent of I_t, it needs to be computed only once.

3 Generalizing Gauss Newton

Now, this is my main idea: I propose that the conventional way of interpreting the SSD GN Hessian (Eq. 7) as a first order approximation of the true Hessian (Eq. 4) is not correct - it should instead be seen as an *estimate of the true Hessian after convergence* since this

interpretation is extensible to other appearance models besides SSD while the other one is not. This idea follows from an examination of the form that Eq. 4 takes if the same substitution $I_t^{\mathbf{w}}(\mathbf{x}) = I_0(\mathbf{x})$ as in Eq. 13 is made to get the corresponding IN Hessian for SSD:

$$\begin{aligned} H_{ssd}^{in} &= \sum_{\mathbf{x}} \left\{ \frac{\partial I_0(\mathbf{x})}{\partial \mathbf{p}} + (I_0 - I_0) \frac{\partial^2 I_0(\mathbf{x})}{\partial \mathbf{p}^2} \right\} \\ &= \left[\nabla I_0(\mathbf{x}) \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \right]^T \left[\nabla I_0(\mathbf{x}) \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \right] \end{aligned} \tag{16}$$

This has the same form as the GN Hessian in Eq. 7 except that the gradient of $I_t^{\mathbf{w}}(\mathbf{x})$ is replaced by that of I_0. In fact, this is the formulation of H_{ssd}^{gn} that is used by the IC formulation of LK. However, the FC formulation uses the form in Eq. 7 and the most straightforward way to convert Eq. 16 to this form is to **replace** I_0 **with** $I_t^{\mathbf{w}}(\mathbf{x})$ instead of the other way around. Let the resultant Hessian be called the **Current Newton (CN)** Hessian or H_{ssd}^{gn} and given as:

$$\begin{aligned} H_{ssd}^{cn} &= \sum_{\mathbf{x}} \left\{ \frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} + (I_t^{\mathbf{w}} - I_t^{\mathbf{w}}) \frac{\partial^2 I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}^2} \right\} \\ &= \left[\nabla I_t^{\mathbf{w}}(\mathbf{x}) \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \right]^T \left[\nabla I_t^{\mathbf{w}}(\mathbf{x}) \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \right] \end{aligned} \tag{17}$$

Since FCLK is known to perform significantly better than ICLK (though this might be partially attributed to the fact that it also uses $I_t^{\mathbf{w}}$ for computing the Jacobian G_{ssd}), one would expect that, if my interpretation holds water, when this substitution is made in Eq. 9 to get the corresponding CN Hessian for MI (Eq. 18), the resultant FC formulation would perform better than that obtained by using the IN Hessian as suggested in [1, 2]. I implemented this and found, as presented in Fig. 1 (a), that FCLK with MI does indeed perform better with CN than with IN on both datasets though the gain in performance is much greater in the more difficult UCSB dataset. It can also be noted that FN performs worst of the three and by quite a wide margin too. The results of GN are not included since, as mentioned before, it inevitably fails in the first frame itself and so has a success rate of zero.

$$\begin{aligned} H_{mi}^{cn} = \sum_{i,j} \Bigg[& \frac{\partial P_{I_t^{\mathbf{w}} I_t^{\mathbf{w}}}(i,j)}{\partial \mathbf{p}}^T \frac{\partial P_{I_t^{\mathbf{w}} I_t^{\mathbf{w}}}(i,j)}{\partial \mathbf{p}} \left(\frac{1}{P_{I_t^{\mathbf{w}} I_t^{\mathbf{w}}}(i,j)} - \frac{1}{P_{I_t^{\mathbf{w}}}(i)} \right) + \\ & \frac{\partial^2 P_{I_t^{\mathbf{w}} I_t^{\mathbf{w}}}(i,j)}{\partial \mathbf{p}^2} \left(1 + \log \left(\frac{P_{I_t^{\mathbf{w}} I_t^{\mathbf{w}}}(i,j)}{P_{I_t^{\mathbf{w}}}(i)} \right) \right) \Bigg] \end{aligned} \tag{18}$$

with

$$\frac{\partial P_{I_t^{\mathbf{w}} I_t^{\mathbf{w}}}(i,j)}{\partial \mathbf{p}} = \frac{1}{n} \sum_{\mathbf{x}} -\frac{\partial \phi}{\partial i} \frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} \phi(j - I_t^{\mathbf{w}}(\mathbf{x})) \tag{19}$$

and

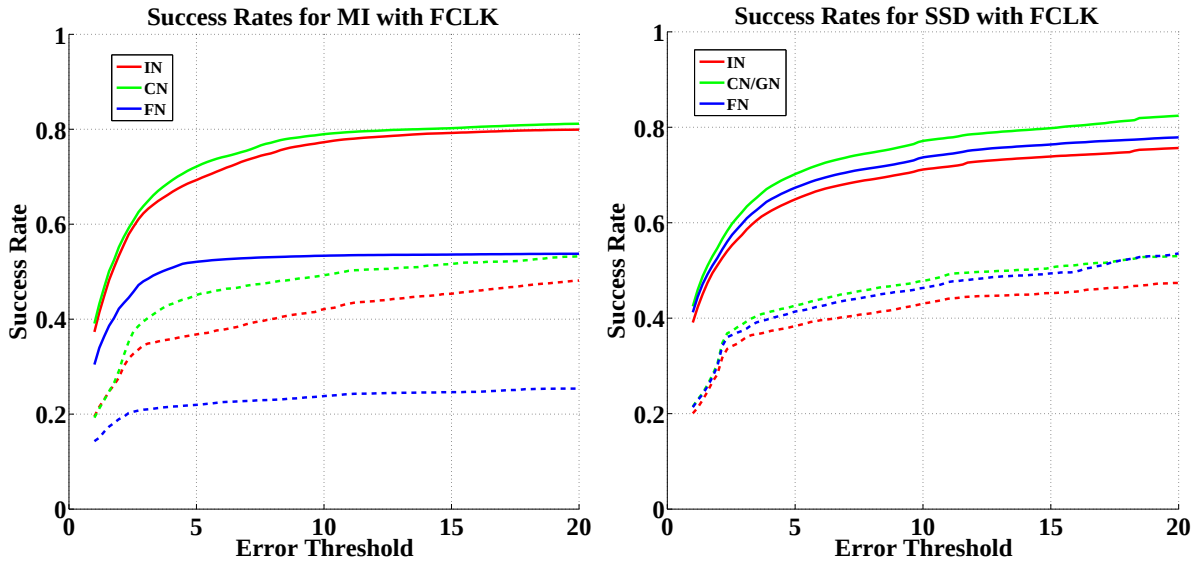
$$\frac{\partial^2 P_{I_t^{\mathbf{w}} I_t^{\mathbf{w}}}(i,j)}{\partial \mathbf{p}^2} = \frac{1}{n} \sum_{\mathbf{x}} \left\{ \left(\frac{\partial^2 \phi}{\partial i^2} \frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}^T \frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} - \frac{\partial \phi}{\partial i} \frac{\partial^2 I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}^2} \right) \phi(j - I_t^{\mathbf{w}}(\mathbf{x})) \right\} \tag{20}$$

The corresponding plots for SSD are shown in part (b) of this figure. CN is significantly better than IN here too but, somewhat surprisingly, FN performs quite well and is in fact

better than IN. This latter result seems rather at odds with the results reported in [7] where the frequency of convergence of FN was shown to be significantly less than that of GN/CN but some difference between static experiments and practical performance is probably to be expected.

Fig.2 presents the performance results for ICLK. The plots for MI are fairly similar to those for FCLK except that IN is a bit closer to CN for TMT probably because of the limiting effect of the initial image Jacobian employed by ICLK. In case of SSD too, IN is very similar to CN for TMT but here FN is almost identical too. There is one rather curious point to be noticed here - the theoretical derivation of the ICLK algorithm involves finding an incremental warp for the *initial* image I_0 that will register it with $I_t^{\mathbf{w}}(\mathbf{x})$ and then updating $\mathbf{p}_t$ with the *inverse* of this warp. As a result, both the Jacobian and the Hessian of the objective function are required to be computed with respect to the initial image (rather than the current one as for FCLK). However, the fact the CN not only works but performs significantly better than IN even though its computation does not involve the initial image at all is, in my opinion, an important indication that the theoretical justifications of the different variants of LK have little to do with their practical workings and they are simply different ways to estimate a single underlying quantity that CN often approximates best.

In addition to the performance plots, I also plotted the shapes of the different Hessians against the perturbations in SSM parameters and found that the relation between the shapes of CN and FN Hessians for MI corresponds very closely to that between GN and FN Hessian for SSD. This can be seen by comparing Figs. 3 and 5 where these plots are shown for translation in x direction. Further analysis of these plots is deferred to Sec. 4 after the remaining Hessians there have been defined.

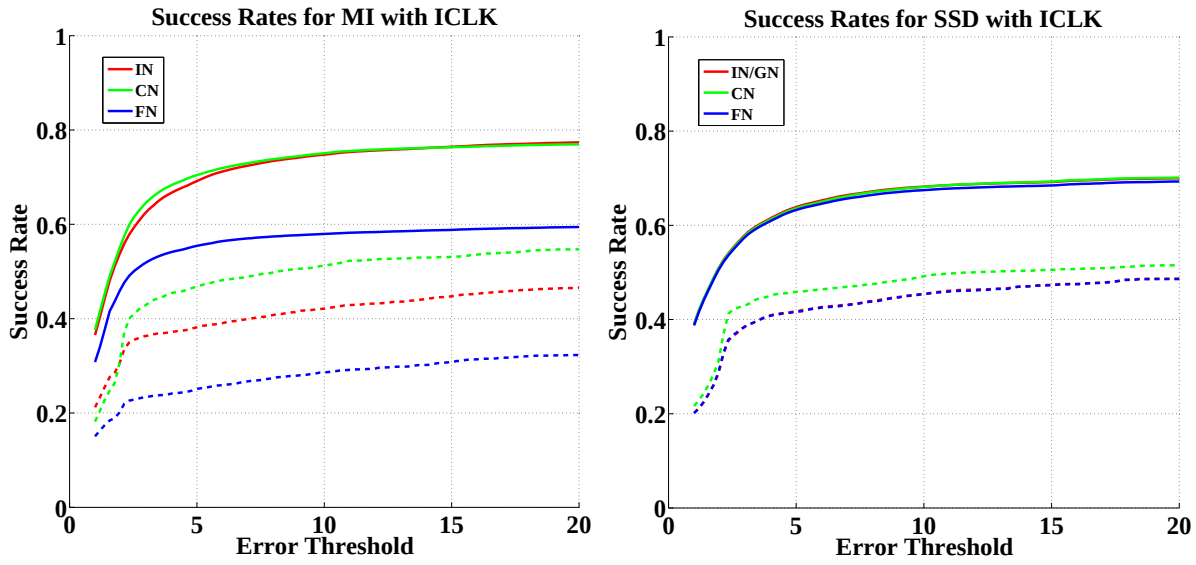


(a) MI (b) SSD

Figure 1: Success rates for SSD and MI using FCLK with IN, CN and FN Hessians. Results over TMT dataset are shown with solid lines while those for UCSB are in dotted lines.

4 ESM

ESM, as defined in its original papers [8, 9], uses the *mean* of the initial and current image gradients to compute the GN Hessian (as well as the Jacobian) that it then uses in Eq. 3 to estimate the incremental update. Let this Hessian be known as the **Mean Newton (MN)**



(a) MI (b) SSD

Figure 2: Success rates for SSD and MI using ICLK with IN, CN and FN Hessians. Results over TMT dataset are shown with solid lines while those for UCSB are in dotted lines.

Hessian. It is given as:

$$H_{ssd}^{mn} = \sum_{\mathbf{x}} \left[\frac{\left(\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} + \frac{\partial I_0(\mathbf{x})}{\partial \mathbf{p}} \right)}{2} \right]^T \left[\frac{\left(\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}} + \frac{\partial I_0(\mathbf{x})}{\partial \mathbf{p}} \right)}{2} \right]$$
$$= \sum_{\mathbf{x}} \left[\left(\frac{\nabla I_t^{\mathbf{w}}(\mathbf{x}) + \nabla I_0(\mathbf{x})}{2} \right) \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \right]^T \left[\left(\frac{\nabla I_t^{\mathbf{w}}(\mathbf{x}) + \nabla I_0(\mathbf{x})}{2} \right) \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \right] \quad (21)$$

At first sight, it would seem that the new interpretation also provides a better way to explain the superior performance of ESM over FCLK as one would intuitively expect that taking the mean of the gradients of initial and current image would provide a better estimate of HAC than either of them used alone (as done in IN and CN respectively). However, a look at the shape of the SSD MN Hessian in Fig. 3 tells a rather different story - this Hessian is actually *not* as good an approximation of HAC as CN, especially farther away from the optimum point.

The plots in Fig.3(a) were generated using frame 10 of the sequence (Fig. 4(a)) before the object had started moving and so looked very similar to the template. As a result, IN seems fairly similar to CN and also very close to HAC. This, however, is not true when the appearance of the object in the current frame differs significantly from that in the initial template due to factors like lighting changes, motion blur or occlusion. An instance of this can be seen in Fig. 3(b) that was generated for frame 167 of the same sequence where significant motion blur is present (Fig. 4(b)). We also see here that MN, as expected, lies somewhere in between IN and CN and so is not as close to HAC as CN. Such a frame with fast motion induced blur is a typical example of a scenario where IN fails but CN continues to track and this difference in their values relative to HAC might well be the reason.

The corresponding plots for MI are shown in Fig. 5. In addition to the five types of Hessians mentioned so far, these figures also show another type called **MN2** which is similar to MN except that the pixel Hessian $\left(\frac{\partial^2 I_t^{\mathbf{w}}}{\partial \mathbf{p}^2} \right)$ too is replaced by the mean over initial and current

images $\left(\frac{\frac{\partial^2 I_t^{\mathbf{w}}}{\partial \mathbf{p}^2} + \frac{\partial^2 I_0}{\partial \mathbf{p}^2}}{2}\right)$. In case of SSD, of course, it is identical to MN as pixel Hessian plays no part in the computation of the self Hessians but it does create a small difference for MI.

Defining the Search Methods

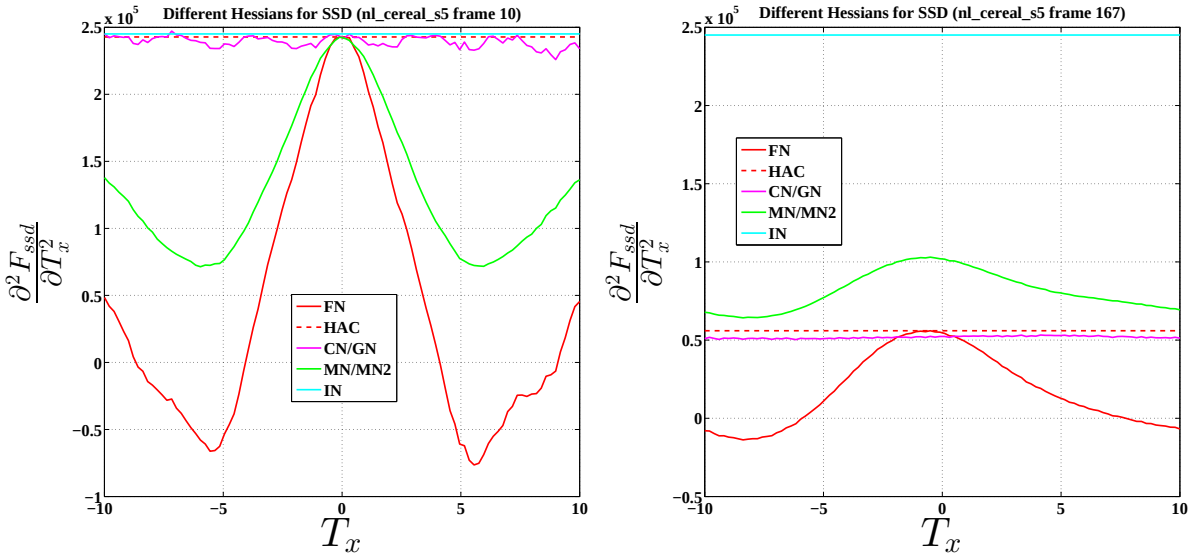
As each of the three search methods - FCLK, ICLK and ESM have been tested with several different types of Hessian, they are defined instead by the form of the Jacobian they employ as follows:

1. **FCLK** : This uses the Jacobian of the similarity function with respect to the current pixel values, i.e.
$$G^{fc} = \frac{\partial F}{\partial I_t^{\mathbf{w}}} \nabla I_t^{\mathbf{w}} \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \tag{22}$$

2. **ICLK** : This uses the Jacobian of the similarity function with respect to the initial pixel values, i.e.
$$G^{ic} = \frac{\partial F}{\partial I_0} \nabla I_0 \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \tag{23}$$
In addition to the form of the Jacobian, another factor that distinguishes ICLK from the other two search methods is that it uses the *inverse* of the incremental warp (computed using Eq. 3) to update the current estimate of the SSM parameters, i.e. it uses the following update equation instead of Eq. 2:
$$\mathbf{p}_t = \mathbf{p}_{t-1} \circ \Delta \mathbf{p}_t^{-1} \tag{24}$$

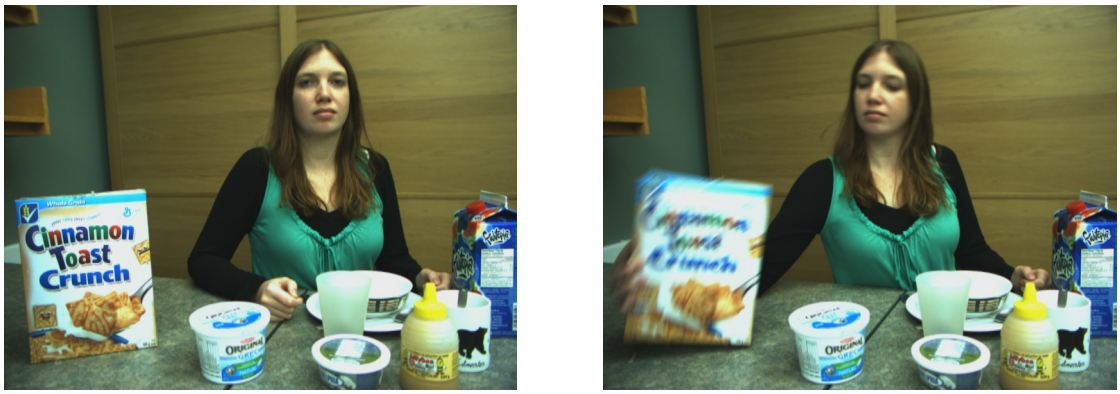
3. **ESM** : As suggested in [10], this uses half the difference between the Jacobians of the similarity function with respect to the initial and current pixel values, i.e.
$$G^{esm} = \frac{\left(\frac{\partial F}{\partial I_0} \nabla I_0 - \frac{\partial F}{\partial I_t^{\mathbf{w}}} \nabla I_t^{\mathbf{w}}\right)}{2} \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \tag{25}$$
This expression may seem strange considering that ESM, in its original formulation [8, 9], was reported as using the *mean* of the gradients. Note, however, that for SSD $\frac{\partial F_{ssd}}{\partial I_0} = (I_0(\mathbf{x}) - I_t^w(\mathbf{x}))$ while $\frac{\partial F_{ssd}}{\partial I_t^{\mathbf{w}}} = (I_t^w(\mathbf{x}) - I_0(\mathbf{x}))$ so that
$$G_{ssd}^{esm} = \frac{(\nabla I_0 + \nabla I_t^{\mathbf{w}})}{2} (I_0(\mathbf{x}) - I_t^w(\mathbf{x})) \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \tag{26}$$
which is same as in [8, 9]. It is, of course, also possible to get this expression using this form instead:
$$G^{mg} = \frac{(\nabla I_0 + \nabla I_t^{\mathbf{w}})}{2} \frac{\partial F}{\partial I_0} \frac{\partial \mathbf{w}}{\partial \mathbf{p}} \tag{27}$$
I tested the performance of ESM with MI using the Jacobians in both Eq. 26 and 27 and found that the former performs better so have used that for all results reported here.

To test if the relative inaccuracy of MN with respect to HAC is also reflected in performance, I generated SR plots for all the different Hessians for ESM too and the results for both MI and SSD are reported in Fig. 6. For MI (Fig. 6(a)), ESM with MN does indeed perform worse than CN (and even IN) for both datasets and by a significant margin too. Note that MN and MN2, though slightly different in appearance (5), gave near identical performance with both datasets. On examining the SSD plots in part (b) of this figure, we can see that, while having a slight performance advantage over CN for TMT, MN is again worse in the more difficult UCSB dataset. In fact even FN is better than MN in this case. The fact that CN outperforms MN in 3 of the 4 tested scenarios even though it cannot be fitted into the theoretical derivation for ESM algorithm reiterates my earlier conjecture that such justifications mean little for practical performance of LK trackers.



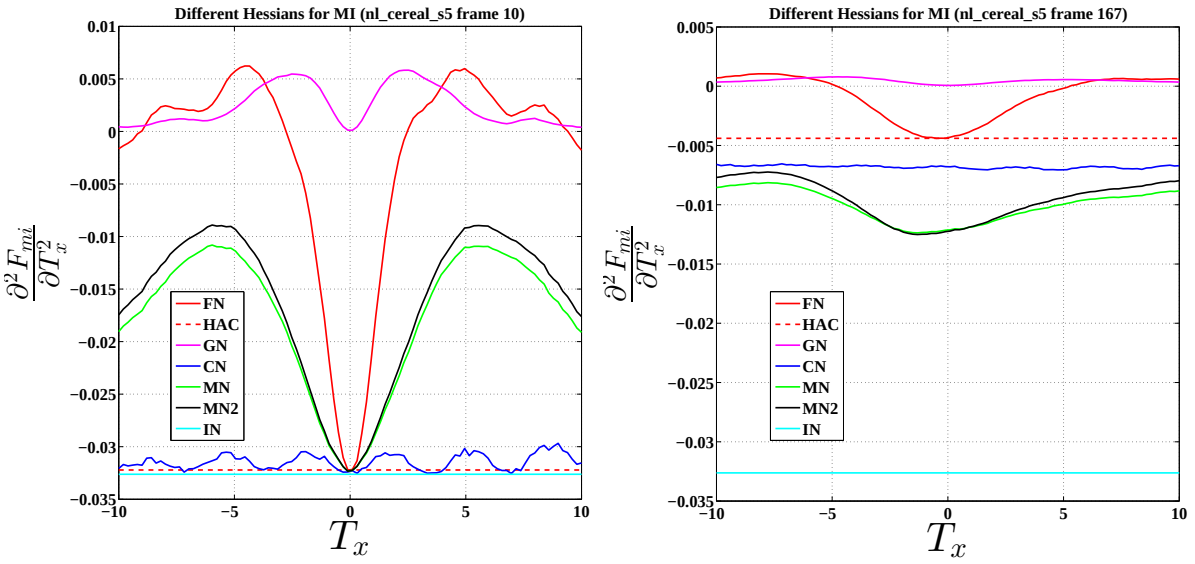
(a) Frame 10 (b) Frame 167

Figure 3: Different Hessian formulations of SSD plotted against horizontal translation for frames 10 and 167 of nl_cereal_s5 sequence from TMT. Zero of the x-axis represents the location of the ground truth in each frame.



(a) Frame 10 (b) Frame 167

Figure 4: Frames from nl_cereal_s5 sequence in TMT dataset



(a) Frame 10 (b) Frame 167

Figure 5: Different Hessian formulations of MI plotted against horizontal translation for frames 10 and 167 of nl_cereal_s5 sequence from TMT. Zero of the x-axis represents the location of the ground truth in each frame.

5 SSIM : Introducing a new Appearance Model

During my meeting with the 501 students about a possible project topic, one of them suggested using **Structural Similarity** (SSIM) [11] as an appearance model in MTF. This is an image similarity index that is very popular for evaluating the quality of image and video compression algorithms by comparing the compressed images with the original ones. In its original form, SSIM is defined as:

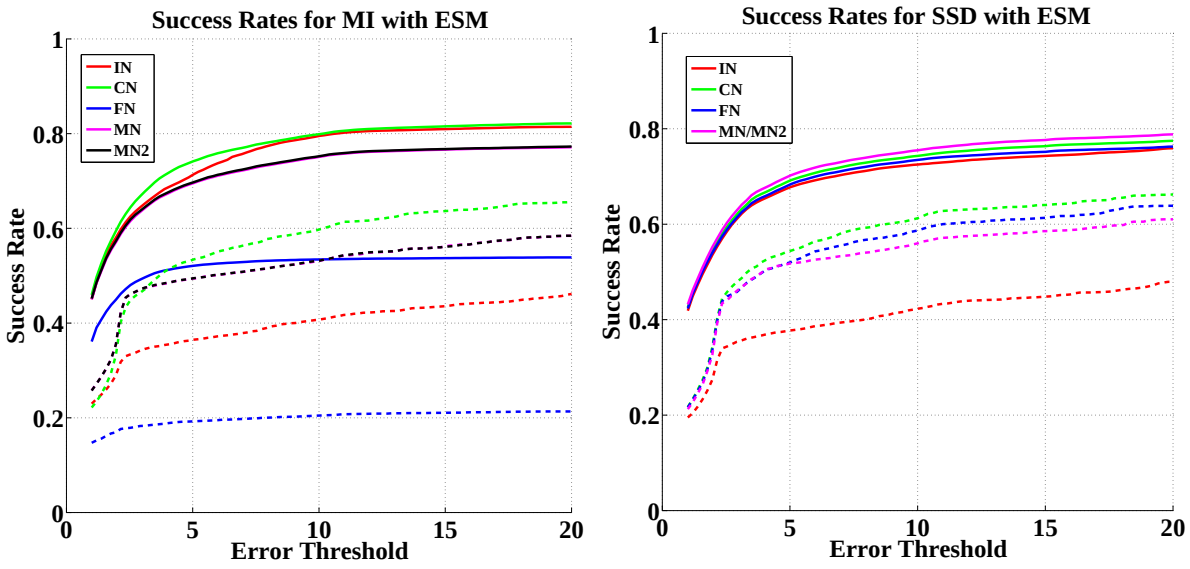
$$F_{ssim}^{orig}(\Delta\mathbf{p}_t) = \sum_{\mathbf{x}} \frac{(2\mu_t(\mathbf{x})\mu_0(\mathbf{x}) + c_1)\,(2\sigma_{t0}(\mathbf{x}) + c_2)}{(\mu_t(\mathbf{x})^2 + \mu_0(\mathbf{x})^2 + c_1)\,(\sigma_t(\mathbf{x})^2 + \sigma_0(\mathbf{x})^2 + c_2)} \tag{28}$$

where $\mu_t(\mathbf{x})$ and $\mu_0(\mathbf{x})$ are the mean pixel values over a neighborhood window (typically 8×8) around $I_t^{\mathbf{w}}(\mathbf{w}(\mathbf{x}, \Delta\mathbf{p}_t))$ and $I_0(\mathbf{x})$, $\sigma_t(\mathbf{x})^2$ and $\sigma_0(\mathbf{x})^2$ are the respective variances, $\sigma_{t0}(\mathbf{x})$ is the covariance and c_1, c_2 are small constants to stabilize division with small denominators. In this form, it is difficult to analytically evaluate the Jacobian and Hessian of F_{ssim} but if the neighborhood window size is reduced to 1×1, the variance and covariance terms disappear and the means are replaced by the pixel values themselves to give a much simpler expression:

$$F_{ssim}(\Delta\mathbf{p}_t) = \sum_{\mathbf{x}} \frac{I_t^{\mathbf{w}}(\mathbf{w}(\mathbf{x}, \Delta\mathbf{p}_t))I_0(\mathbf{x})}{I_t^{\mathbf{w}}(\mathbf{w}(\mathbf{x}, \Delta\mathbf{p}_t))^2 + I_0(\mathbf{x})^2} \tag{29}$$

where c_1 and multiplication by 2 have been dropped for added simplicity. Though the original formulation of SSIM has been used before for tracking using particle filters [12] and gradient descent [13], to my knowledge this simplified version has never been used. Also, existing formulations have only used it for 2 DOF tracking probably due to the aforementioned complexity of evaluating its derivatives. In the simplified form, the analytical Jacobian and Hessian become quite simple to compute:

$$\frac{\partial F_{ssim}}{\partial I_t^{\mathbf{w}}} = \sum_{\mathbf{x}} \frac{I_0(\mathbf{x})[I_0(\mathbf{x})^2 - I_t^{\mathbf{w}}(\mathbf{x})^2]}{[I_0(\mathbf{x})^2 + I_t^{\mathbf{w}}(\mathbf{x})^2]^2} \tag{30}$$



(a) MI (b) SSD

Figure 6: Success rates for SSD and MI using ESM with IN, CN, FN, MN and MN2 Hessians. Results over TMT dataset are shown with solid lines while those for UCSB are in dotted lines.

$$\frac{\partial^2 F_{ssim}}{(\partial I_t^{\mathbf{w}})^2} = \sum_{\mathbf{x}} \frac{2I_0(\mathbf{x})I_t^{\mathbf{w}}(\mathbf{x})[I_t^{\mathbf{w}}(\mathbf{x})^2 - 3I_0(\mathbf{x})^2]}{[I_0(\mathbf{x})^2 + I_t^{\mathbf{w}}(\mathbf{x})]^3} \tag{31}$$

The two self Hessians CN and IN are respectively given as:

$$\frac{\partial^2 F_{ssim}^{cn}}{(\partial I_t^{\mathbf{w}})^2} = -\sum_{\mathbf{x}} \frac{1}{2I_t^{\mathbf{w}}(\mathbf{x})^2} \tag{32}$$

$$\frac{\partial^2 F_{ssim}^{in}}{(\partial I_0)^2} = -\sum_{\mathbf{x}} \frac{1}{2I_0(\mathbf{x})^2} \tag{33}$$

Combining these equations with the standard chain rule for second order derivatives, expressions for all the Hessians can be obtained as follows:

$$H_{ssim}^{fn} = \sum_{\mathbf{x}} \left\{ \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}\right]^T \frac{\partial^2 F_{ssim}}{(\partial I_t^{\mathbf{w}})^2} \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}\right] + \frac{\partial F_{ssim}}{\partial I_t^{\mathbf{w}}} \frac{\partial^2 I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}^2} \right\} \tag{34}$$

$$H_{ssim}^{gn} = \sum_{\mathbf{x}} \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}\right]^T \frac{\partial^2 F_{ssim}}{(\partial I_t^{\mathbf{w}})^2} \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}\right] \tag{35}$$

$$H_{ssim}^{cn} = \sum_{\mathbf{x}} \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}\right]^T \frac{\partial^2 F_{ssim}^{cn}}{(\partial I_t^{\mathbf{w}})^2} \left[\frac{\partial I_t^{\mathbf{w}}(\mathbf{x})}{\partial \mathbf{p}}\right] \tag{36}$$

$$H_{ssim}^{in} = \sum_{\mathbf{x}} \left[\frac{\partial I_0(\mathbf{x})}{\partial \mathbf{p}}\right]^T \frac{\partial^2 F_{ssim}^{in}}{(\partial I_0)^2} \left[\frac{\partial I_0(\mathbf{x})}{\partial \mathbf{p}}\right] \tag{37}$$

I plotted the objective function and its Jacobian to see if they are smooth enough for good convergence properties and the results, shown in Fig. 7 are quite promising and in fact resemble MI quite closely in spite of there being no relation between their objective functions. The plots

of the different Hessians (Fig. 8) too are fairly similar to those for MI (Fig. 5) though perhaps with a bit more jitter, probably due to the absence of the stabilizing constant. One difference that can be noted in Fig. 8 is that GN is much closer to CN which is rather reminiscent of SSD. It may also be noted that, like SSD, MN is equal to MN2 here too. Given that its analytical form is closer to SSD while its plots resemble those of MI instead, I decided to implement and test this model to check if the observations reported for SSD and MI are extensible to this too.

The performance results with FCLK and ICLK are shown in Fig. 9. It can be seen that CN is indeed better than all other Hessians for all cases except one - FCLK with UCSB dataset for some reason shows unexpectedly poor performance with CN so that it is just at par with FN and GN. This is in stark contrast to ICLK where CN leads by a wide margin. In fact, ICLK actually performs better than FCLK with CN in UCSB which is very unusual since ICLK often fares poorly in this dataset as it contains a lot of sequences with fast motion. This apparent disparity can also be observed, albeit to a lesser degree, by comparing the IC and FC plots for SSD (Figs. 1b and 2b). For TMT, however, the relative performances are very similar to that for MI (Figs. 1a and 2a) with CN leading IN by a modest margin and FN and GN being far behind.

Fig. 10 shows the results with ESM where again patterns from both SSD and MI are apparent. Similar to SSD (Fig. 6b), MN has a small lead over CN in TMT but lags far behind in UCSB where it is even worse than FN. However, similar to MI (Fig. 6a), FN fares very poorly in TMT and is the worst performer by a significant margin. One final point that can be noted from Fig. 9 is that, as expected from the greater similarity in the shapes of CN and GN (Fig. 8), the latter performs quite well here (compared to MI) and is usually at par with FN.

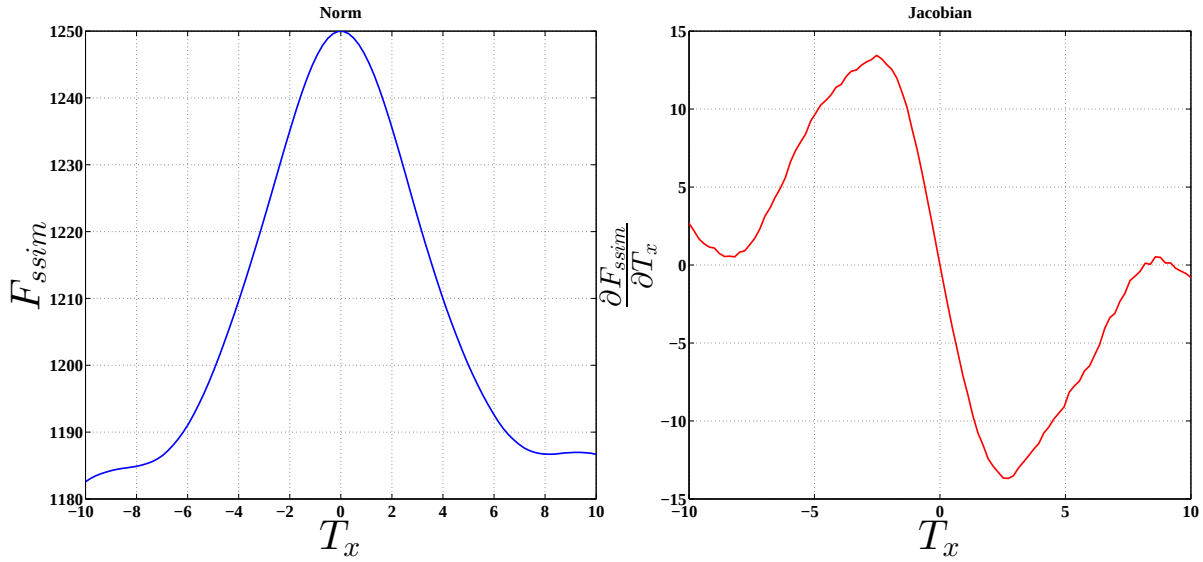
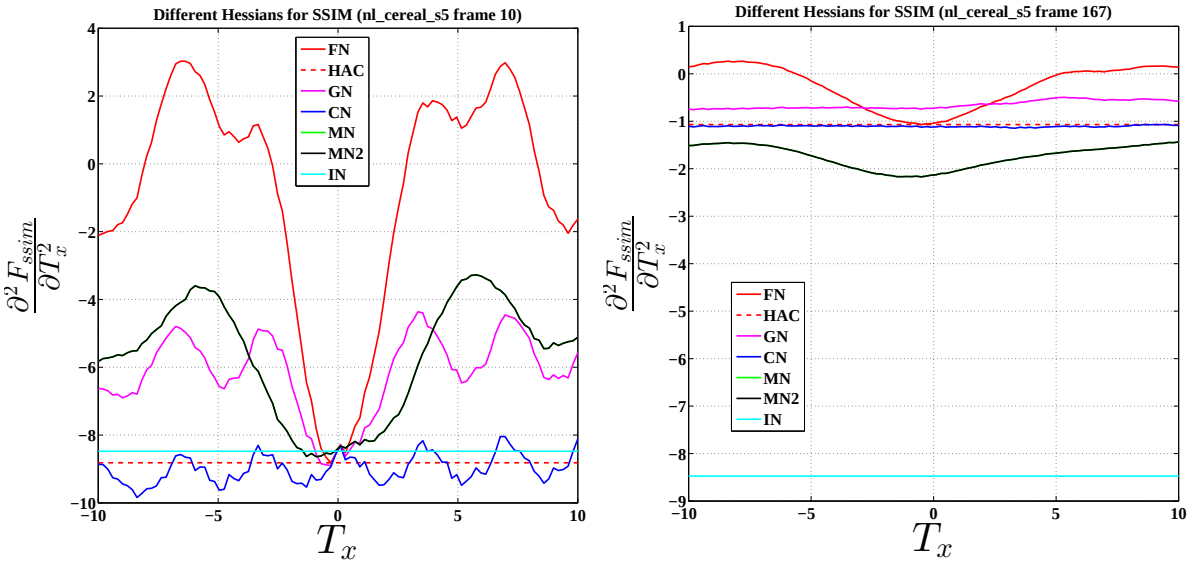


Figure 7: SSIM objective function and its Jacobian plotted against horizontal translation

6 Conclusions

I have presented a new way to interpret the workings of Lucas Kanade type trackers in this report along with (hopefully) substantial of evidence in support of my hypothesis. In addition to MI and SSD, I have also presented results for a new appearance model SSIM to strengthen my case. If my proposal turns out to be correct, this may provide a new direction along which further research may be directed to improve the performance of Lucas Kanade type trackers. I

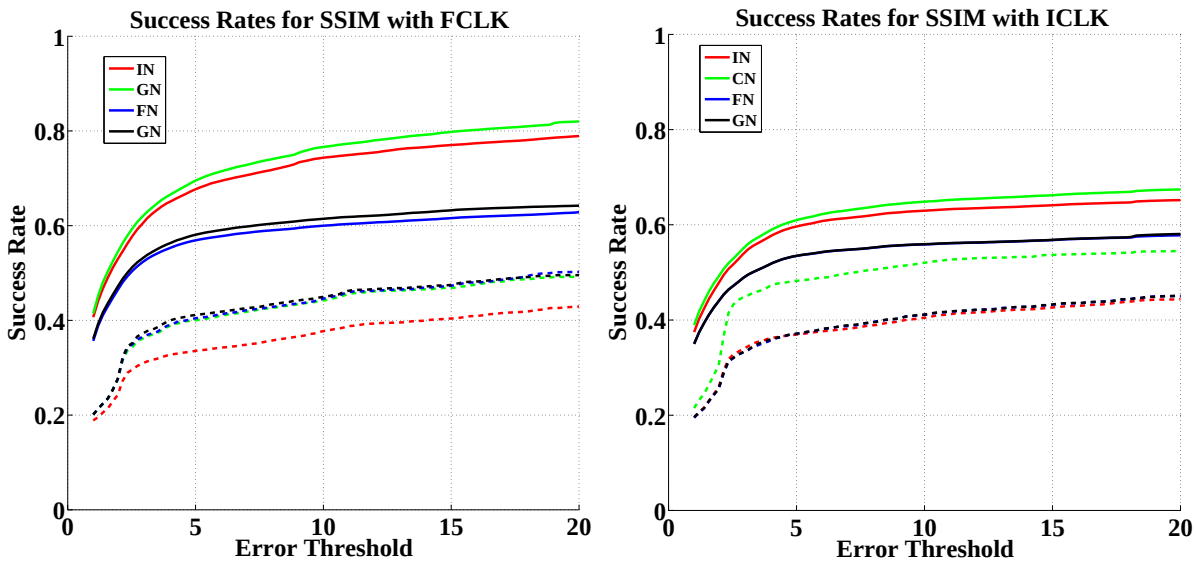


(a) Frame 10

(b) Frame 167

Figure 8: Different Hessian formulations of SSIM plotted against horizontal translation for frames 10 and 167 of nl_cereal_s5 sequence from TMT. Zero of the x-axis represents the location of the ground truth in each frame.

have thought up a few ways that the ground truth can be used to get an improved estimate of HAC and thus test this theory better but would like your opinion first.



(a) FCLK (b) ICLK

Figure 9: Success rates for SSIM using FCLK and ICLK with the different Hessian types. Results over TMT dataset are shown with solid lines while those for UCSB are in dotted lines.

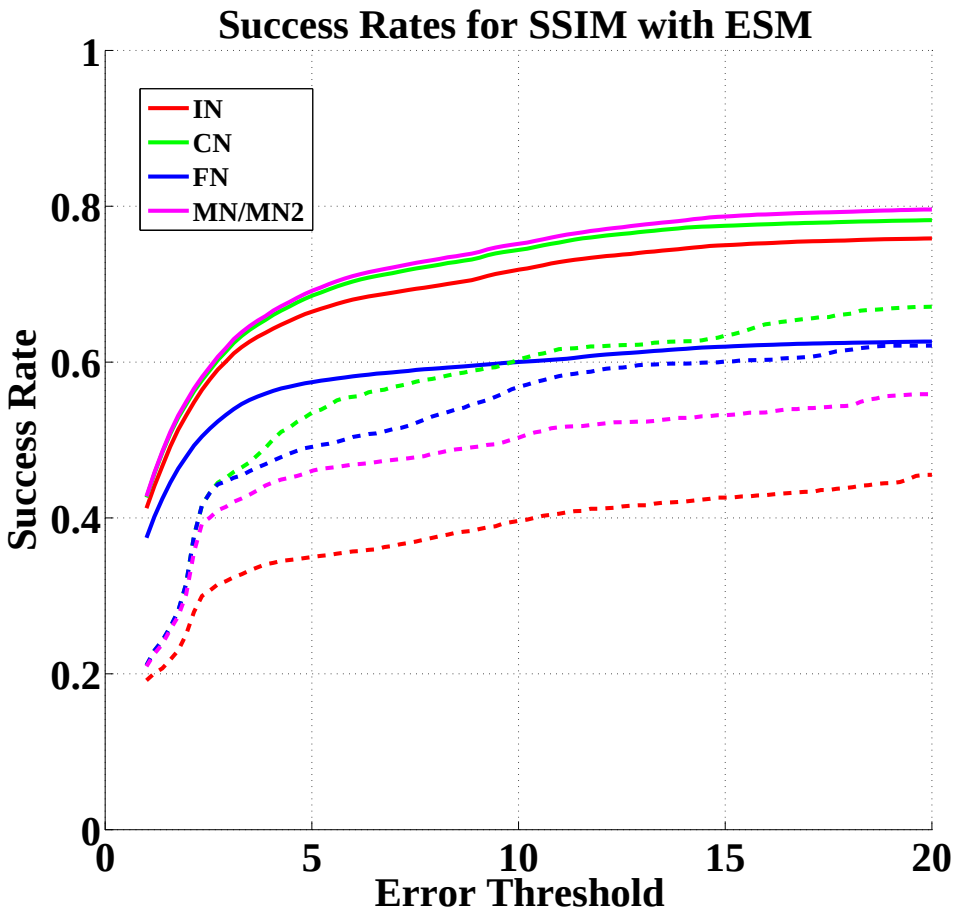


Figure 10: Success rates for SSIM using ESM with the different Hessian types. Results over TMT dataset are shown with solid lines while those for UCSB are in dotted lines.

References

[1] Amaury Dame. *A unified direct approach for visual servoing and visual tracking using mutual information*. PhD thesis, University of Rennes, 2010.

[2] Amaury Dame and Eric Marchand. Accurate real-time tracking using mutual information. In *IEEE International Symposium on Mixed and Augmented Reality*, pages 47–56, 2010.

[3] Nicholas Dowson and Richard Bowden. Mutual Information for Lucas-Kanade Tracking (MILK): An Inverse Compositional Formulation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(1):180–185, Jan 2008.

[4] P. Thevenaz and M. Unser. Optimization of mutual information for multiresolution image registration. *Image Processing, IEEE Transactions on*, 9(12):2083–2099, Dec 2000.

[5] Nicholas Dowson and Richard Bowden. A Unifying Framework for Mutual Information Methods for Use in Non-linear Optimisation. In Ale Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 365–378. Springer Berlin Heidelberg, 2006.

[6] GaussNewton algorithm. https://en.wikipedia.org/wiki/Gauss%E2%80%93Newton_algorithm. Accessed: 2015-11-26.

[7] Simon Baker and Iain Matthews. Lucas-Kanade 20 Years On: A Unifying Framework. *International Journal of Computer Vision*, 56(3):221–255, Feb 2004.

[8] Selim Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 943–948 vol.1, Sept 2004.

[9] S. Benhimane and E. Malis. Homography-based 2D Visual Tracking and Servoing. *Int. J. Rob. Res.*, 26(7):661–676, July 2007.

[10] Rupert Brooks and Tal Arbel. Generalizing Inverse Compositional and ESM Image Alignment. *International Journal of Computer Vision*, 87(3):191–212, May 2010.

[11] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *Image Processing, IEEE Transactions on*, 13(4):600–612, April 2004.

[12] A. Loza, L. Mihaylova, N. Canagarajah, and David Bull. Structural Similarity-Based Object Tracking in Video Sequences. In *Information Fusion, 2006 9th International Conference on*, pages 1–6, July 2006.

[13] A. Loza, Fanglin Wang, Jie Yang, and L. Mihaylova. Video object tracking with differential Structural SIMilarity index. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 1405–1408, May 2011.