CVPR Supplementary Material For Submission 1391

Anonymous Submission

1 Soft vs. Hard Constraint

The term "weak constraint" refers to the following definition of alpha:

$$\alpha_{weak} = \frac{1}{mn^2}. \tag{1}$$

The term "strong constraint" refers to the following definition of alpha:

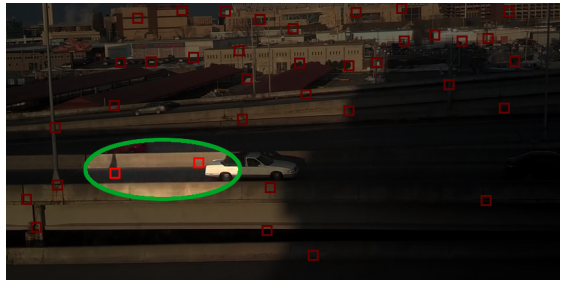
$$\alpha_{strong} = \frac{1}{mFn^2}. \tag{2}$$

Each definition includes a constant m to adjust the relative strength of the penalty term in the energy function. With the weak constraint α is defined in such a way that the penalty term is on the same scale as a single fit term. This is done so that a single strong feature can overpower the penalty, but a weak feature cannot. This enables tracking in environments where the low rank constraint is only approximately satisfied (since strong features can ignore the penalty). The penalty is effectively only used to help with low-quality features.

With the strong constraint, the penalty term is on the same scale as the total contribution of all fit terms. With this definition, it requires the consensus of a significant fraction of the features in a scene to overpower the penalty term. This is only appropriate when one expects all features in the scene to strictly obey a low rank model. Deviations from the model are considered errors. The hard constraint allows tracking through limited occlusion, since a majority of features in the scene can coerce a small number of occluded features to ignore bad template matches and instead move in a way that maintains a low rank trajectory set. Figure 1 illustrates the difference between the two constraints.

(a) Occluded Features in Frame 1

(b) Occluded Features in Frame 10 - Weak Constraint



(c) Occluded Features in Frame 10 - Strong Constraint

Fig. 1: A scene with many features from a rigid environment and 2 strong features occluded by a moving vehicle. Images (b) and (c) show the results of tracking with the weak and strong constraints, respectively. With the weak constraint, the features track the moving vehicle. With the strong constraint, the features ignore the vehicle and track the occluded environment. (Best viewed in color)

2 Selection of m For Experiments

The experimental results with the weak constraint require a value of m to be chosen to set the relative strength of the penalty term. The procedure for this was to take the first video in the set of test videos and for each penalty form, run the tracker on this one video with the penalty term effectively disabled (very small m). We then iteratively

increased m and re-ran the tracker. We increased m until the penalty was so strong that the fit terms were insignificant and tracking no longer worked. We then identified the value that gave the best results on that first video. When a range of values seamed to give nearly the same (best) results, we selected the midpoint of that range. The resulting values that were used for the results shown in the paper were:

Constraint Form	m
Empirical Dimension - Uncentered $\boldsymbol{M}$	0.1
Empirical Dimension - Centered $\boldsymbol{M}$	0.15
Nuclear Norm - Uncentered $\boldsymbol{M}$	0.001
Nuclear Norm - Centered $\boldsymbol{M}$	0.0005
Explicit Factorization - Uncentered $\boldsymbol{M}$	0.0015
Explicit Factorization - Centered $\boldsymbol{M}$	0.002

3 Algorithm Complexity

A single iteration of our optimization algorithm requires one evaluation of the gradient of the energy function $\bar{C}$ and multiple evaluations of $\bar{C}$ (without gradient). To compute the gradient of a fit term, we could use the chain rule to express the derivative in terms of the image gradient of our frame, I. However, each fit term depends on only two coordinates of $\mathbf{x}$. Hence, with only 4 evaluations of a fit term we can get a direct numerical estimate of it's derivative. We have found that this yields a more accurate result than using the chain rule and image gradients. An evaluation of a fit term requires collecting and processing pixel intensities (or RGB intensities) from n^2 locations, and we have F fit terms. Thus, the number of operations required to evaluate all fit terms and their collective contributions to the gradient of $\bar{C}$ is proportional to Fn^2.

To compute the gradient of any of our penalty terms, the main computational expense is computing the SVD of the $2(L+1)$-by-F matrix $\boldsymbol{M}$. Without employing any tricks, this requires on the order of $LF^2 + L^3 + F^2L$ operations. Since L will generally be a small constant, this is effectively order LF^2.

Thus, a complete gradient computation has a complexity of $k_1Fn^2 + k_2LF^2$. Energy evaluation only requires each fit term to be evaluated one time and the singular values (but not the full decomposition) of $\boldsymbol{M}$. This has complexity $k_3Fn^2 + k_4L^2F$. The number of plain energy evaluations depends on tolerances set in the optimization algorithm, so it is useful to have actual benchmarks for performance. As mentioned in the paper, our single-threaded implementation can run on modest hardware with real-time performance (35 features of size 7-by-7 can be tracked with $L = 5$ at 16 frames per second, or with $L = 10$ at several frames per second).

4 Qualitative Tests on Genuine Low Quality Video

Included in this submission of supplementary material is a folder called "Qualitative Experiments on Real Videos". This folder contains multiple sub-directories with video sequences that were not synthetically degraded. In each sub-directory, the original video is included, along with versions with overlayed results of 3 different trackers. Results are presented for Lucas Kanade, 1^{st}-order descent (no constraints), and a representative result of our constrained tracker (we present the results with the empirical dimension penalty form and un-centered $\mathbf{M}$). We do not include compiled videos for all trackers due to the size restrictions on the supplemental material.

5 Videos for Quantitative Experiments

Included in this submission of supplementary material is a folder called "Quantitative Test Videos". This contains subdirectories with the source test videos and tracker output and results files for all trackers referred to in the paper. The results and output files follow a naming convention and each tracker is referred to using a unique ID. The following is the mapping between IDs and tracker names/descriptions.

ID	Tracker Name/Description
0	KLT
2	1st-Order Descent
5	Multi-Tracker - Emp Dim - Uncentered
6	Multi-Tracker - Emp Dim - Centered
7	Multi-Tracker - Nuc Norm - Uncentered
8	Multi-Tracker - Nuc Norm - Centered
9	Multi-Tracker - Exp Fact - Uncentered
10	Multi-Tracker - Exp Fact - Centered
100	LDOF

Output trajectory files follow a simple format. Each line contains the trajectory of a single feature in the form (f_1,x_1,y_1):(f_2,x_2,y_2):... where f_n is a frame number, and (x_n, y_n) is the position in image coordinates of the feature in frame f_n. Positions are measured in pixels and x is vertical position, while y is horizontal position (both measured from the top left corner). Each output trajectory file has either "mode1" or "mode2" in its filename. "mode1" refers to the test where features are initialized in frame 0 and never re-initialized (each feature is tracked until wandering off-screen). "mode2" refers to the test where features are re-initialized when they wander more than 10 pixels from ground truth (and mean time between re-initializations is used as the performance metric).

For convenience, we include videos of each source video with results overlayed from 3 of the trackers used in this comparison. We include videos for KLT, 1^{st}-order descent (no constraint), and representative results for our constrained tracker (we present the results with the empirical dimension penalty form and un-centered $\mathbf{M}$). In the paper we present results of 2 different experiments (referred to above). It is not easy to

visually present the results of the second experiments (since poorly tracked feature are immediately re-initialized with ground truth). We therefore only include video overlays for the first set of experiments (where each tracker is run on the same set of features for 30 frames).

6 Additional Quantitative Experiments

We include (in the folder "Quantitative Test Videos - Additional") an additional set of quantitative tests on another set of videos. The videos in this set are much shorter (only 30 frames) and we present mean drift and mean L1 trajectory error after 30 frames. Since the videos are much shorter, we do not present results for the other experiment conducted in the paper (where features are re-initialized when wandering 10 or more pixels from ground truth and mean time between re-initializations is measured). LDOF was not run on this set of test videos.

The videos in this set were recorded using a different camera with a different resolution than the tests in the main paper. These videos were also degraded in a different fashion (non-uniform darkening and directional motion blur were added). Thus, the strength coefficients (m) for the different penalty forms needed to be re-evaluated (two of the penalty forms are sensitive to the video resolution because normalized coordinates were not used). The coefficients were selected in the same manner as for the test set in the main paper, by tuning each tracker on the first video in the set. The resulting values used for this set of videos were:

Constraint Form	m
Empirical Dimension - Uncentered $\boldsymbol{M}$	0.5
Empirical Dimension - Centered $\boldsymbol{M}$	0.4
Nuclear Norm - Uncentered $\boldsymbol{M}$	0.004
Nuclear Norm - Centered $\boldsymbol{M}$	0.002
Explicit Factorization - Uncentered $\boldsymbol{M}$	0.05
Explicit Factorization - Centered $\boldsymbol{M}$	0.05

Again, we include videos with tracker output overlayed for convenience. Results from the same 3 trackers as the other tests are included.

Table 1: Mean feature drift after 30 frames of tracking

		Video Number										Average
		1	2	3	4	5	6	7	8	9	10	
Tracker	KLT	58.6	158.9	59.7	203.3	178.4	91.8	156.6	71.6	148.0	77.5	120.4
	1st-Order Descent	29.1	13.8	39.8	18.6	85.7	39.9	10.9	9.5	15.9	59.8	32.3
	Multi-Tracker - Emp Dim - Uncentered	14.1	11.1	19.1	3.8	60.3	22.9	11.6	3.3	10.6	17.2	17.4
	Multi-Tracker - Emp Dim - Centered	**9.4**	11.4	15.3	5.9	60.0	**12.5**	11.2	3.1	**8.6**	19.6	**15.7**
	Multi-Tracker - Nuc Norm - Uncentered	13.3	11.0	**14.5**	**3.5**	62.7	22.4	16.7	4.2	11.2	**16.5**	17.6
	Multi-Tracker - Nuc Norm - Centered	13.2	**6.9**	17.8	3.6	59.9	21.0	**8.2**	**2.6**	11.7	28.9	17.4
	Multi-Tracker - Exp Fact - Uncentered	18.8	25.5	25.9	8.6	67.6	23.8	16.4	5.3	19.8	41.4	25.3
	Multi-Tracker - Exp Fact - Centered	13.3	18.3	16.2	7.9	**55.5**	13.0	16.3	4.4	16.5	21.0	18.2

Table 2: Mean L1 trajectory error after 30 frames of tracking

		Video Number										Average
		1	2	3	4	5	6	7	8	9	10	
Tracker	KLT	1414.2	2739.0	1155.7	4534.9	2585.6	1684.4	2399.8	1579.9	2212.0	1350.5	2165.6
	1st-Order Descent	418.1	286.6	542.7	409.6	1010.8	495.0	298.2	152.5	260.1	950.4	482.4
	Multi-Tracker - Emp Dim - Uncentered	180.5	165.8	286.0	137.5	654.7	269.7	176.3	**63.1**	179.4	**330.2**	244.3
	Multi-Tracker - Emp Dim - Centered	**131.7**	173.3	**199.4**	129.8	600.0	**139.6**	**161.0**	63.7	**128.7**	389.5	**211.7**
	Multi-Tracker - Nuc Norm - Uncentered	186.2	155.3	210.5	134.1	703.6	271.1	231.7	75.5	190.5	333.4	249.2
	Multi-Tracker - Nuc Norm - Centered	175.9	**147.4**	243.3	**118.6**	687.6	222.5	170.0	63.4	189.0	474.4	249.2
	Multi-Tracker - Exp Fact - Uncentered	274.6	435.9	410.4	218.5	753.0	377.8	269.5	96.0	343.7	750.0	392.9
	Multi-Tracker - Exp Fact - Centered	185.5	287.8	216.3	147.2	**543.8**	153.1	228.5	79.3	275.7	433.5	255.1

7 Gradient Evaluation

The gradients of template fit terms (all but the final term in Eq. (3)) are evaluated numerically using the centered derivative estimate (sampling 0.25 pixels up, down, left, and right of the given location). We found this to give better accuracy than using the chain rule and image gradients. We evaluate the gradient of the penalty term $P(\mathbf{x})$ in Eq. (3) explicitly, so the precise formula depends on the form of P. In the following descriptions, we occasionally use Matlab notation to more clearly express some operations. Each gradient computation has a similar structure:

- Construct the centered or uncentered matrix $\boldsymbol{M}$ (centering is done by subtracting the mean trajectory from each column).
- Evaluate the SVD of $\boldsymbol{M}$: $\boldsymbol{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$.
- Define $\frac{d\mathrm{d}_{\mathrm{dim}}}{d\boldsymbol{M}} = \mathbf{U}\tilde{\mathbf{\Sigma}}\mathbf{V}^T$, where $\tilde{\mathbf{\Sigma}}$ is defined in the following subsections for each regularizer.
- Let $\mathbf{A}$ be the matrix containing only the final 2 rows of $\frac{d\mathrm{d}_{\mathrm{dim}}}{d\boldsymbol{M}}$.
- Let $\mathbf{B}$ be the 2-by-1 vector containing the means of the two rows of $\mathbf{A}$.
- In the centered case, let $\mathbf{C} = \mathbf{A} - \mathbf{B}$ ones(1,F). In the uncentered case, let $\mathbf{C} = \mathbf{A}$.
- Let $\overrightarrow{\mathbf{C}} = \mathbf{C}(:)$. That is, vectorize $\mathbf{C}$ by taking one column at a time.
- $\nabla P = \overrightarrow{\mathbf{C}}^T$.

7.1 Nuclear Norm

For the nuclear norm low-rank regularizer: $P(\mathbf{x}) = \|\boldsymbol{M}\|_* = \|\boldsymbol{\sigma}\|_1$. We compute the gradient ∇P as described above. For this regularizer, $\tilde{\mathbf{\Sigma}}$ is diagonal with $\tilde{\mathbf{\Sigma}}_{i,i} = 1$ if $\mathbf{\Sigma}_{i,i} > \xi$ and $\tilde{\mathbf{\Sigma}}_{i,i} = \mathbf{\Sigma}_{i,i}/\xi$ otherwise. Here, ξ is a small number (we use 0.05).

This reduces sensitivity to small singular values and generally yields a better search direction.

7.2 Explicit Factorization

Recall that the explicit factorization regularizer is $P(\mathbf{x}) = \sum_{i=d+1}^{F} \sigma_i$, where σ_i is the i'th singular value of $\boldsymbol{M}$. The gradient computations for the centered and uncentered cases are described above. For this regularizer, $\tilde{\boldsymbol{\Sigma}}$ is diagonal with $\tilde{\boldsymbol{\Sigma}}_{i,i} = 0$ for $i \in \{1, 2, ..., d\}$. For $i > d$, $\tilde{\boldsymbol{\Sigma}}_{i,i} = 1$ if $\boldsymbol{\Sigma}_{i,i} > \xi$ and $\tilde{\boldsymbol{\Sigma}}_{i,i} = \boldsymbol{\Sigma}_{i,i}/\xi$ otherwise. As with the nuclear norm regularizer, we use $\xi = 0.05$.

7.3 Empirical Dimension

Recall that the empirical dimension regularizer is $P(\mathbf{x}) = \hat{d}_\epsilon(\boldsymbol{M}) := \frac{\|\boldsymbol{\sigma}\|_\epsilon}{\|\boldsymbol{\sigma}\|_{\left(\frac{\epsilon}{1-\epsilon}\right)}}$. The gradient computations for the centered and uncentered cases are described above. Let $\delta = \epsilon/(1-\epsilon)$. Now let $C_1 = \|\boldsymbol{\sigma}\|_\epsilon^{1-\epsilon}\|\boldsymbol{\sigma}\|_\delta^{-1}$ and $C_2 = \|\boldsymbol{\sigma}\|_\epsilon\|\boldsymbol{\sigma}\|_\delta^{-1-\delta}$. For this regularizer, $\tilde{\boldsymbol{\Sigma}}$ is diagonal with $\tilde{\boldsymbol{\Sigma}}_{i,i} = \left(C_1\sigma_i^{\epsilon-1} - C_2\sigma_i^{\delta-1}\right)$ if $\left(C_1\sigma_i^{\epsilon-1} - C_2\sigma_i^{\delta-1}\right) > \xi$, and $\tilde{\boldsymbol{\Sigma}}_{i,i} = (\sigma_i/\xi)\left(C_1\sigma_i^{\epsilon-1} - C_2\sigma_i^{\delta-1}\right)$ otherwise. Again, we use $\xi = 0.05$.