Real-Time Video Stabilization for Handheld Devices

Dimo Dimov, Assos. Prof., IICT-BAS, Bulgaria
Atanas Nikolov, Ph.D. student, IICT-BAS, Bulgaria
1. Introduction

- Video stabilization seeks to create a stable version of casually shot video (usually filmed on a handheld device, such as a mobile phone or a portable camcorder) which is typically shaky and undirected, i.e. it suffers from all the disadvantages of a non-stationary camera filming.

- By contrast, professional cinematographers carefully plan camera motion along smooth simple paths, using a wide variety of sophisticated equipment, such as tripods, cameras mounted on rails, camera dollies, steadicams, etc. Such hardware is impractical for many situations (or expensive for amateurs), so the software video stabilization is a widely-used for improving casual video.
1. Introduction

- In principle, the task of software 2D stabilization (and even 3D stabilization) is considered solved in the cases of an off-line processing, and even in real time, but if using a powerful enough computer and a parallel implementation on the GPU.

- However, at least for now, the computing power of mobile phones is not enough to achieve acceptable video stabilization and therefore it is relied also on the usage of the inertial sensors (gyroscopes and/or accelerometers) in the phone’s hardware.

- The method proposed here concerns the 2D video stabilization and aims at achieving minimal implementation complexity in a combination with enough precision, so that it can be used as a periodical initializing contrivance for the system of inertial sensors in the mobile phone, finally responsible for the full (possibly 3D) video stabilization in real time.
2.1. 2D Video Stabilization: pros (+) and cons (-)

- In general, the methods for 2D video stabilization are based on the estimation of an optimal linear transformation of the current frame to a reference one (e.g. previous frame) in a given video sequence. The actual stabilization is realized through a transformation, inverse to the calculated one that converts (translates, rotates, scales, etc.) the current frame to achieve an optimal match to the reference one.

- In case of approximately plain scenes (with an arbitrary camera movement) or cases where the camera shake is strictly rotational (within an arbitrary scene), unwanted jitters can be effectively reduced based on two-dimensional reasoning of the video.

- Assuming the scene geometry and camera motion do fall into these categories, such 2D stabilization methods are **robust**, operate on the **entire frame** and consume **minimal computing efforts**.

...
2.1. 2D Video Stabilization: pros (+) and cons (-)

• However, most scenes do contain objects at arbitrary depths and in many scenarios, such as hand-held camera shoots, it is actually impossible to avoid any translational component in the camera shake.

• In these cases, a full-frame matching cannot model the parallax that is induced by a translational shift in viewpoint and this level of scene modelling is insufficient for video stabilization.

• The second limitation of these 2D motion models is that there is no knowledge of the 3D trajectory of the input camera, making it impossible to simulate an idealized camera path similar to what can be found in professional tracking shots.
2.2. 3D Video Stabilization in Brief

- The first realization of 3D video stabilization for dynamic scenes was described in the paper: Content-Preserving Warps for 3D Video Stabilization (Liu et al. 2009).

- But, despite achieved high quality of movement smoothness there, its application was practically limited by the need of carrying out a 3D reconstruction of scenes through SFM (Structure from Motion) method.

- SFM has restrictions towards robustness and generality because some videos simply do not contain sufficient motion information to allow for reconstruction. These restrictions are:

  1. Parallax (because of depth of the real scene);
  2. Camera zooming;
  3. In-camera stabilization (the img sensor instead the lenses);
  4. Rolling shutter (of CMOS performances, but not for CCD).
  5. Other (lenses, atmospheric blur, in-scene motions, ...)

Beyond this restrictions, the processing speed was also a significant problem there, because SFM requires a global nonlinear optimization.

- Because of the above mentioned problems with 3D video stabilization, a trend has appeared recently toward the usage of 2D approaches, because of their computational efficiency and robustness.
2.3. Interesting Methods for 2D Video Stabilization
2.3. Interesting Methods for 2D Video Stabilization

- …See, the paper; Eq.1.

- …Because of this, we call their modeling transformation: ‘approximated matrix model’.

- In contrast, we offer graphically clear ‘an accurate vector model’ of changes between successive frames, which also allows for finding of the elementary geometric transformations – translation, rotation, scale, skew, etc.

- Simple approaches to motion vectors evaluation:
  - unregular distribution of characteristic points (SfM) <-> frames regular division on subimages;
  - directly from MPEG: not very good idea, because of only for basic frames;
  - via 2D differences for each couple of consecutive frames:
    - in L2 => (Euclidian distance) => 2D correlation => usage of FFT to speed up
    - in L1 => (Manhattan): SAD (Sum of Absolute Differences): better but slow
  - Resp. approximations via horizontal & vertical histograms (sum of intensities): …
3. Description of Our Method for Video Stabilization

3.1. Evaluation of the Motion Vectors

Before considering the Accurate Vector Model (AVM) of our method, we will give a brief description of the basic scheme for determining of the so called ‘motion vectors’, similarly to the TI developments, but with our improvements.
3. Description of Our Method for Video Stabilization

3.1. Evaluation of the Motion Vectors (2)

Our improvement consists of the following: instead of the conventional $\text{IPH}_j^k$ and $\text{IPV}_j^k$, we use in the above calculation the normalized projections, $\overline{\text{IPH}}_j^k$ and $\overline{\text{IPV}}_j^k$, i.e. centralized by their average, moreover - the average is a ‘floating’ one. For example, along horizontals we have:

$$\overline{\text{IPH}}_j^k(x, \tau) = \text{IPH}_j^k(x) - \overline{\text{IPH}}_j^k(\tau), \quad \overline{\text{IPH}}_j^k(\tau) = \frac{1}{x_{wsiz}} \sum_{p=-x_{wsiz}/2}^{x_{wsiz}/2} \text{IPH}_j^k(\tau + p),$$

as well as by analogy – along verticals.

Similarly, as in the papers of TI, the respective 9 motion vectors $\vec{t}_{ji} = (t_{xji}^{(k)}, t_{yji}^{(k)})$ for the k-th frame, are estimated as a minimum by the SAD approach:

$$t_{xji}^{(k)} = \arg\min_{\tau} \{ \text{SADH}_j^k(\tau) \}; \quad \text{SADH}_j^k(\tau) = \sum_{x=-x_{wsiz}/2}^{x_{wsiz}/2} |\text{IPH}_j^k(x + \tau) - \text{IPH}_j^{k-1}(x)|, \quad -t_H < \tau < t_H;$$

$$t_{yji}^{(k)} = \arg\min_{\tau} \{ \text{SADV}_j^k(\tau) \}; \quad \text{SADV}_j^k(\tau) = \sum_{y=-y_{wsiz}/2}^{y_{wsiz}/2} |\text{IPV}_j^k(y + \tau) - \text{IPV}_j^{k-1}(y)|, \quad -t_V < \tau < t_V.$$
3. Description of Our Method for Video Stabilization
3. Description of Our Method for Video Stabilization

3.2. Description of the Accurate Vector Model (AVM) (2)

Thereby, we form 9 vector equations, one for each pair of centres \((C_{ji}^k, C_{ji}^{k-1})\), \((j = 0,1,2), (i = 0,1,2)\) in given two (not necessarily consecutive) frames \((k - 1)\) and \((k)\), \(k = 1,2,...\) from the video clip. The unknown parameters are: the translation vector \(\vec{T} = (T_x, T_y)\), which is one and the same \((\vec{T}_{ji} = \vec{T})\) for each pair of centres; and the reference rotation vector \(\vec{r} = (r_x, r_y)\), which is chosen to be \(\vec{r} = \vec{r}_{12} = (r_{x12}, r_{y12})\), and by which the remaining rotation vectors \(\vec{r}_{ji}\) are expressed (for one and the same angle \(\alpha\)). Thus, we compose a system of 18 component equations for the 4 unknowns \((T_x, T_y, r_x, r_y)\), and we find them by LSM, see Table 1.

<table>
<thead>
<tr>
<th>Centre ((j,i))</th>
<th>LSM on (\partial x)</th>
<th>(\partial \varepsilon_{xji}^2 / 2\partial r_x)</th>
<th>(\partial \varepsilon_{xji}^2 / 2\partial r_y)</th>
<th>LSM on (\partial y)</th>
<th>(\partial \varepsilon_{yji}^2 / 2\partial r_x)</th>
<th>(\partial \varepsilon_{yji}^2 / 2\partial r_y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>((1,1))</td>
<td>(T_x - t_{x11} = \varepsilon_{x11})</td>
<td>0</td>
<td>0</td>
<td>(T_y - t_{y11} = \varepsilon_{y11})</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>((1,2))</td>
<td>(T_x + r_x - t_{x12} = \varepsilon_{x12})</td>
<td>(\varepsilon_{x12})</td>
<td>0</td>
<td>(T_y + r_y - t_{y12} = \varepsilon_{y12})</td>
<td>0</td>
<td>(\varepsilon_{y12})</td>
</tr>
<tr>
<td>((1,0))</td>
<td>(T_x - r_x - t_{x10} = \varepsilon_{x10})</td>
<td>(-\varepsilon_{x10})</td>
<td>0</td>
<td>(T_y - r_y - t_{y10} = \varepsilon_{y10})</td>
<td>0</td>
<td>(-\varepsilon_{y10})</td>
</tr>
<tr>
<td>((2,1))</td>
<td>(T_x - kr_y - t_{x21} = \varepsilon_{x21})</td>
<td>0</td>
<td>(-k\varepsilon_{x21})</td>
<td>(T_y + kr_x - t_{y21} = \varepsilon_{y21})</td>
<td>(k\varepsilon_{y21})</td>
<td>0</td>
</tr>
<tr>
<td>((0,1))</td>
<td>(T_x + kr_y - t_{x01} = \varepsilon_{x01})</td>
<td>0</td>
<td>(k\varepsilon_{x01})</td>
<td>(T_y - kr_x - t_{y01} = \varepsilon_{y01})</td>
<td>(-k\varepsilon_{y01})</td>
<td>0</td>
</tr>
<tr>
<td>((2,2))</td>
<td>(T_x + r_x - kr_y - t_{x22} = \varepsilon_{x22})</td>
<td>(\varepsilon_{x22})</td>
<td>(-k\varepsilon_{x22})</td>
<td>(T_y + kr_x + r_y - t_{y22} = \varepsilon_{y22})</td>
<td>(k\varepsilon_{y22})</td>
<td>(\varepsilon_{y22})</td>
</tr>
<tr>
<td>((0,0))</td>
<td>(T_x - r_x + kr_y - t_{x00} = \varepsilon_{x00})</td>
<td>(-\varepsilon_{x00})</td>
<td>(k\varepsilon_{x00})</td>
<td>(T_y - kr_x - r_y - t_{y00} = \varepsilon_{y00})</td>
<td>(-k\varepsilon_{y00})</td>
<td>(-\varepsilon_{y00})</td>
</tr>
<tr>
<td>((0,2))</td>
<td>(T_x + r_x + kr_y - t_{x02} = \varepsilon_{x02})</td>
<td>(\varepsilon_{x02})</td>
<td>(k\varepsilon_{x02})</td>
<td>(T_y - kr_x + r_y - t_{y02} = \varepsilon_{y02})</td>
<td>(-k\varepsilon_{y02})</td>
<td>(\varepsilon_{y02})</td>
</tr>
<tr>
<td>((2,0))</td>
<td>(T_x - r_x - kr_y - t_{x20} = \varepsilon_{x20})</td>
<td>(-\varepsilon_{x20})</td>
<td>(-k\varepsilon_{x20})</td>
<td>(T_y + kr_x - r_y - t_{y20} = \varepsilon_{y20})</td>
<td>(k\varepsilon_{y20})</td>
<td>(-\varepsilon_{y20})</td>
</tr>
</tbody>
</table>
3. Description of Our Method for Video Stabilization

3.2. Description of the Accurate Vector Model (AVM) (3)

Besides the classical matrix approach (as far as the case of LSM is linear), the searched optimal solution for \((T_x, T_y, r_x, r_y)\), can be calculated by the global minimum of the objective function \(L = \sum_{j,i} |\tilde{e}_{ji}|^2 = \sum_{j,i} (e_{xji}^2 + e_{yji}^2)\), namely by solving a system of 4 equations by zeroing the respective first partial derivatives \(\frac{\partial L}{\partial T_x}\), \(\frac{\partial L}{\partial T_y}\), \(\frac{\partial L}{\partial r_x}\) and \(\frac{\partial L}{\partial r_y}\) (in case of non-negativity of the second partial derivatives). Under this more general approach, the following direct solution can be obtained:

\[
T_x = \frac{1}{9} \sum_{j,i} t_{xji}, \quad T_y = \frac{1}{9} \sum_{j,i} t_{xji};
\]

\[
r_x = \frac{(t_{x12} - t_{x10} + t_{x22} - t_{x00} + t_{x02} - t_{x20}) + k(t_{y21} - t_{y01} + t_{y22} - t_{y00} - t_{y02} + t_{y20})}{6 + 6k^2},
\]

\[
r_y = \frac{(t_{y12} - t_{y10} + t_{y22} - t_{y00} + t_{y02} - t_{y20}) - k(t_{x21} - t_{x01} + t_{x22} - t_{x00} - t_{x02} + t_{x20})}{6 + 6k^2}, \quad \alpha = 2 \arctg(r_x / r_y);
\]

which is more efficient for real time use than the classical matrix approach for linear LSM.
3. Description of Our Method for Video Stabilization

3.3. Improvements of the Proposed Basic Algorithm

To increase the accuracy of the found solution we apply LSM second time, but on a smaller number of equations for the searched unknowns ($T_x$, $T_y$, $r_x$, $r_y$). For this purpose, it is enough to introduce new parameters $e_{ji}$, $(j = 0,1,2)$, $(i = 0,1,2)$, as follows: $e_{ji} = 0$, if the equation $(j, i)$ will be eliminated, otherwise $e_{ji} = 1$. Thus, for the second LSM, we get a modified system of 18 equations, see Table 2.

<table>
<thead>
<tr>
<th>Centre ($j,i$)</th>
<th>LSM on $\vec{O}x$</th>
<th>$e_{xji}$</th>
<th>$\frac{\partial \varepsilon_{xji}^2}{2 \partial r_x}$</th>
<th>LSM on $\vec{O}y$</th>
<th>$e_{yji}$</th>
<th>$\frac{\partial \varepsilon_{yji}^2}{2 \partial r_x}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,1)</td>
<td>$(T_x - t_{x11} = \varepsilon_{x11})e_{x11}$</td>
<td>0</td>
<td>0</td>
<td>$(T_y - t_{y11} = \varepsilon_{y11})e_{y11}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(1,2)</td>
<td>$(T_x + r_x - t_{x12} = \varepsilon_{x12})e_{x12}$</td>
<td>$e_{x12}e_{x12}$</td>
<td>0</td>
<td>$(T_y + r_y - t_{y12} = \varepsilon_{y12})e_{y12}$</td>
<td>0</td>
<td>$e_{y12}e_{y12}$</td>
</tr>
<tr>
<td>(1,0)</td>
<td>$(T_x - r_x - t_{x10} = \varepsilon_{x10})e_{x10}$</td>
<td>$-e_{x10}e_{x10}$</td>
<td>0</td>
<td>$(T_y - r_y - t_{y10} = \varepsilon_{y10})e_{y10}$</td>
<td>0</td>
<td>$-e_{y10}e_{y10}$</td>
</tr>
<tr>
<td>(2,1)</td>
<td>$(T_x - kr_y - t_{x21} = \varepsilon_{x21})e_{x21}$</td>
<td>0</td>
<td>$-ke_{x21}e_{x21}$</td>
<td>$(T_y + kr_x - t_{y21} = \varepsilon_{y21})e_{y21}$</td>
<td>$ke_{y21}e_{y21}$</td>
<td>0</td>
</tr>
<tr>
<td>(0,1)</td>
<td>$(T_x + kr_y - t_{x01} = \varepsilon_{x01})e_{x01}$</td>
<td>0</td>
<td>$ke_{x01}e_{x01}$</td>
<td>$(T_y - kr_x - t_{y01} = \varepsilon_{y01})e_{y01}$</td>
<td>$-ke_{y01}e_{y01}$</td>
<td>0</td>
</tr>
<tr>
<td>(2,2)</td>
<td>$(T_x + r_x - kr_y - t_{x22} = \varepsilon_{x22})e_{x22}$</td>
<td>$e_{x22}e_{x22}$</td>
<td>$-ke_{x22}e_{x22}$</td>
<td>$(T_y + kr_x + r_y - t_{y22} = \varepsilon_{y22})e_{y22}$</td>
<td>$ke_{y22}e_{y22}$</td>
<td>$e_{y22}e_{y22}$</td>
</tr>
<tr>
<td>(0,0)</td>
<td>$(T_x - r_x + kr_y - t_{x00} = \varepsilon_{x00})e_{x00}$</td>
<td>$-e_{x00}e_{x00}$</td>
<td>$ke_{x00}e_{x00}$</td>
<td>$(T_y - kr_x - r_y - t_{y00} = \varepsilon_{y00})e_{y00}$</td>
<td>$-ke_{y00}e_{y00}$</td>
<td>$-e_{y00}e_{y00}$</td>
</tr>
<tr>
<td>(0,2)</td>
<td>$(T_x + r_x + kr_y - t_{x02} = \varepsilon_{x02})e_{x02}$</td>
<td>$e_{x02}e_{x02}$</td>
<td>$ke_{x02}e_{x02}$</td>
<td>$(T_y - kr_x + r_y - t_{y02} = \varepsilon_{y02})e_{y02}$</td>
<td>$-ke_{y02}e_{y02}$</td>
<td>$e_{y02}e_{y02}$</td>
</tr>
<tr>
<td>(2,0)</td>
<td>$(T_x - r_x - kr_y - t_{x20} = \varepsilon_{x20})e_{x20}$</td>
<td>$-e_{x20}e_{x20}$</td>
<td>$-ke_{x20}e_{x20}$</td>
<td>$(T_y + kr_x + r_y - t_{y20} = \varepsilon_{y20})e_{y20}$</td>
<td>$ke_{y20}e_{y20}$</td>
<td>$-e_{y20}e_{y20}$</td>
</tr>
</tbody>
</table>

Table 2. LSM equations of the extended AVM.
### 3. Description of Our Method for Video Stabilization

#### 3.3. Improvements of the Proposed Basic Algorithm

In the proposed method, we assume a Gaussian distribution of the error offsets, $e_{xji}$ and $e_{yji}$, $(j = 0,1,2)$, $(i = 0,1,2)$, obtained at the first LSM pass. Thus, we evaluate 18 coefficients, of the type $e_{xji}$ and $e_{yji}$, as follows:

$$e_{xji} = \begin{cases} 0, & \text{if } |e_{xji}| > \eta \bar{e}_x \\ 1, & \text{if } |e_{xji}| \leq \eta \bar{e}_x \end{cases}$$

and similarly for $e_{yji}$, where $\bar{e}_x$ and $\bar{e}_y$ are the mean square deviations $(\bar{e}_x)^2 = \frac{1}{9} \sum_{j,i} (e_{xji} - \bar{e}_x)^2$ and $(\bar{e}_y)^2 = \frac{1}{9} \sum_{j,i} (e_{yji} - \bar{e}_y)^2$, towards corresponding mean values, $\bar{e}_x$ and $\bar{e}_y$; and $\eta$ is an expert coefficient, $1 < \eta < 3$. The primary idea is to eliminate the equations, giving relatively large deviations at the first LSM pass.

<table>
<thead>
<tr>
<th>Centre $(j,i)$</th>
<th>LSM on $\vec{O}x$</th>
<th>$e_{xji} \frac{\partial e_{xji}^2}{2\partial r_x}$</th>
<th>$e_{xji} \frac{\partial e_{xji}^2}{2\partial r_y}$</th>
<th>LSM on $\vec{O}y$</th>
<th>$e_{yji} \frac{\partial e_{yji}^2}{2\partial r_x}$</th>
<th>$e_{yji} \frac{\partial e_{yji}^2}{2\partial r_y}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(1,1)$</td>
<td>$(T_x - t_{x1} = e_{x11})e_{x11}$</td>
<td>0</td>
<td>0</td>
<td>$(T_y - t_{y1} = e_{y11})e_{y11}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$(1,2)$</td>
<td>$(T_x + r_x - t_{x12} = e_{x12})e_{x12}$</td>
<td>$e_{x12}e_{x12}$</td>
<td>0</td>
<td>$(T_y + r_y - t_{y12} = e_{y12})e_{y12}$</td>
<td>0</td>
<td>$e_{y12}e_{y12}$</td>
</tr>
<tr>
<td>$(1,0)$</td>
<td>$(T_x - r_x - t_{x10} = e_{x10})e_{x10}$</td>
<td>$-e_{x10}e_{x10}$</td>
<td>0</td>
<td>$(T_y - r_y - t_{y10} = e_{y10})e_{y10}$</td>
<td>0</td>
<td>$-e_{y10}e_{y10}$</td>
</tr>
<tr>
<td>$(2,1)$</td>
<td>$(T_x - kr_y - t_{x21} = e_{x21})e_{x21}$</td>
<td>0</td>
<td>$-ke_{x21}e_{x21}$</td>
<td>$(T_y + kr_x - t_{y21} = e_{y21})e_{y21}$</td>
<td>$ke_{y21}e_{y21}$</td>
<td>0</td>
</tr>
<tr>
<td>$(0,1)$</td>
<td>$(T_x + kr_y - t_{x01} = e_{x01})e_{x01}$</td>
<td>0</td>
<td>$ke_{x01}e_{x01}$</td>
<td>$(T_y - kr_x - t_{y01} = e_{y01})e_{y01}$</td>
<td>$-ke_{y01}e_{y01}$</td>
<td>0</td>
</tr>
<tr>
<td>$(2,2)$</td>
<td>$(T_x + r_x - kr_y - t_{x22} = e_{x22})e_{x22}$</td>
<td>$e_{x22}e_{x22}$</td>
<td>$-ke_{x22}e_{x22}$</td>
<td>$(T_y + kr_x + r_y - t_{y22} = e_{y22})e_{y22}$</td>
<td>$ke_{y22}e_{y22}$</td>
<td>$e_{y22}e_{y22}$</td>
</tr>
<tr>
<td>$(0,0)$</td>
<td>$(T_x - r_x + kr_y - t_{x00} = e_{x00})e_{x00}$</td>
<td>$-e_{x00}e_{x00}$</td>
<td>$ke_{x00}e_{x00}$</td>
<td>$(T_y - kr_x - r_y - t_{y00} = e_{y00})e_{y00}$</td>
<td>$-ke_{y00}e_{y00}$</td>
<td>$-e_{y00}e_{y00}$</td>
</tr>
<tr>
<td>$(0,2)$</td>
<td>$(T_x + r_x + kr_y - t_{x02} = e_{x02})e_{x02}$</td>
<td>$e_{x02}e_{x02}$</td>
<td>$ke_{x02}e_{x02}$</td>
<td>$(T_y + kr_x + r_y - t_{y02} = e_{y02})e_{y02}$</td>
<td>$-ke_{y02}e_{y02}$</td>
<td>$e_{y02}e_{y02}$</td>
</tr>
<tr>
<td>$(2,0)$</td>
<td>$(T_x - r_x - kr_y - t_{x20} = e_{x20})e_{x20}$</td>
<td>$-e_{x20}e_{x20}$</td>
<td>$-ke_{x20}e_{x20}$</td>
<td>$(T_y + kr_x + r_y - t_{y20} = e_{y20})e_{y20}$</td>
<td>$ke_{y20}e_{y20}$</td>
<td>$-e_{y20}e_{y20}$</td>
</tr>
</tbody>
</table>

**Table 2. LSM equations of the extended AVM.**
4. Experimental Analysis

For a quantitative analysis of our algorithms quality, we use Interframe Transformation Fidelity (ITF) metrics:

\[
\text{ITF} = \frac{1}{N - 1} \sum_{k=1}^{N-1} \text{PSNR}((k), (k - 1)),
\]

where PSNR is a peak signal-to-noise ratio between two consecutive frames \((k - 1)\) and \((k)\), and \(N\) is the number of frames in the video clip. The PSNR measures the degree of similarity between two images, which makes it suitable for the evaluation of the proximity between frames:

\[
\text{PSNR}((k), (k - 1)) = 10 \log_{10} \left( \text{MSE}((k), (k - 1)) \right), \text{ where}
\]

\[
\text{MSE}((k), (k - 1)) = \frac{1}{x_{\text{size}} y_{\text{size}}} \left( 1 + \sum_{y=0}^{y_{\text{size}}-1} \sum_{x=0}^{x_{\text{size}}-1} \text{diff}(I_k(x, y), I_{k-1}(x, y)) \right),
\]

where \(I_k(x, y)\) is the value of pixel \((x, y)\) in \((k)\)-th frame, and

\[
\text{diff}(a, b) = \begin{cases} 
1, & \text{if } |a - b| > t \\
0, & \text{otherwise}
\end{cases}, \quad 0 \leq a, b \leq 255,
\]

where \(t\) is a threshold, \(t \leq 255\).

We define experimentally the threshold value \(t = 32\), so that it is suitable for our test videos. The aim is ITF to give sufficiently different results between an original and stabilized video and between videos, stabilized by the different methods compared here.
4. Experimental Analysis

Experiments have been conducted on 5 video clips of frame size (640x480), with a different degree of jittering, assessed on a scale – from 1 (for low) to 5 (for strong), see Fig. 3. The respective ITF values for the quality of the experimented approaches by our method for stabilization, in these test videos, are shown in Table 3. The experiments are C++ and/or Matlab written in Windows 7, and are carried out on an Intel Q9550 CPU 2.83 GHz, RAM 12GB, HDD 7200 rpm.

Fig.3: Tested video clips ordered increasingly by their degree of shaking (a frame is only shown per clip).

Table 3. ITF measures for 5 video-clips: the originals and their stabilization by 6 approaches.

<table>
<thead>
<tr>
<th>video</th>
<th>method</th>
<th>original</th>
<th>3x3/noAVG/I</th>
<th>3x3/AVG/I</th>
<th>3x3/noAVG/II</th>
<th>3x3/AVG/II</th>
<th>9x9/AVG/II</th>
<th>FAST</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>11.85</td>
<td>16.57</td>
<td>17.45</td>
<td>17.68</td>
<td>17.99</td>
<td>17.99</td>
<td>18.09</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>11.09</td>
<td>14.36</td>
<td>16.06</td>
<td>16.01</td>
<td>16.43</td>
<td>16.29</td>
<td>16.37</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>10.01</td>
<td>14.02</td>
<td>14.79</td>
<td>14.84</td>
<td>15.30</td>
<td>15.51</td>
<td>15.70</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>9.61</td>
<td>13.05</td>
<td>13.66</td>
<td>13.85</td>
<td>14.03</td>
<td>14.31</td>
<td>14.32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AVM [ms]</th>
<th>Totals [ms] [fps]</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>33</td>
</tr>
<tr>
<td>6</td>
<td>109</td>
</tr>
<tr>
<td>5</td>
<td>111</td>
</tr>
<tr>
<td>5</td>
<td>110</td>
</tr>
<tr>
<td>7</td>
<td>141</td>
</tr>
<tr>
<td>37</td>
<td>156</td>
</tr>
<tr>
<td>(52)</td>
<td></td>
</tr>
</tbody>
</table>
4. Experimental Analysis

Table 3. ITF measures for 5 video-clips: the originals and their stabilization by 6 approaches.

<table>
<thead>
<tr>
<th>video</th>
<th>original</th>
<th>3x3/noAVG</th>
<th>3x3/AVG</th>
<th>3x3/noAVG</th>
<th>3x3/AVG</th>
<th>9x9/AVG</th>
<th>FAST</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11.85</td>
<td>16.57</td>
<td>17.45</td>
<td>17.68</td>
<td>17.99</td>
<td>17.99</td>
<td>18.09</td>
</tr>
<tr>
<td>2</td>
<td>11.09</td>
<td>14.36</td>
<td>16.06</td>
<td>16.01</td>
<td>16.43</td>
<td>16.29</td>
<td>16.37</td>
</tr>
<tr>
<td>3</td>
<td>10.01</td>
<td>14.02</td>
<td>14.79</td>
<td>14.84</td>
<td>15.30</td>
<td>15.51</td>
<td>15.70</td>
</tr>
<tr>
<td>4</td>
<td>9.61</td>
<td>13.05</td>
<td>13.66</td>
<td>13.85</td>
<td>14.03</td>
<td>14.31</td>
<td>14.32</td>
</tr>
<tr>
<td>AVM [ms]</td>
<td>-</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>7</td>
<td>37</td>
<td>(52)</td>
</tr>
<tr>
<td>Totals [ms]</td>
<td>33</td>
<td>111</td>
<td>109</td>
<td>111</td>
<td>110</td>
<td>141</td>
<td>156</td>
</tr>
<tr>
<td>[fps]</td>
<td>30</td>
<td>9.02</td>
<td>9.18</td>
<td>9.00</td>
<td>9.09</td>
<td>7.09</td>
<td>6.41</td>
</tr>
</tbody>
</table>

For the C++ version, the pure durations of our method are given on row 6 of Table 3. Subtracting them from the totals (see row 7), the rest obviously dominate and can be shared out on average as follows: for read \(\approx 15\) ms, for image reverse transform \(\approx 75\) ms, for save \(\approx 10\) ms, and for extra operations \(\approx 5\) ms. Thus, the achieved maximal speed (in fps) is \(\approx 7\div 9\) fps. We assume these outside operations can be executed quite more efficiently on the special hardware of a handheld device; e.g. \(\approx 3\) times faster will be enough to achieve the conventional speed of \(\approx 30\) fps.
4. Experimental Analysis

The following conclusions can also be drawn based on the experiment results:

♦ ITF evaluations correspond to the degree of camera shaking, and we believe it is adequate to our visual classification (i.e., the higher ITF ⇔ the better stabilized video);

♦ The worst ITF results are obtained by (3x3/noAVG/I)-approach, i.e. when neither IP (integral projections) normalization nor the second LSM pass (LSM-II) is used;

♦ The (3x3/noAVG/II)-stabilization is almost as unacceptable as the (3x3/AVG/II)-one, but according to the estimated ITF, it follows that LSM-II has a higher impact on the output result quality, rather than the IP normalization;

♦ The quality of results (visually and by ITF) is significantly increased at (3x3/AVG/II), i.e. at a combination of both, the IP normalization and the LSM-II;

♦ Usage of more detailed division scheme causes (compare (9x9/AVG/II) and (3x3/AVG/II)) better stabilization of/on videos of low quality (less ITF) than of/on better quality videos (high ITF);

♦ The most pleasant results, as expected, have been obtained by the (FAST) method, and it is well visible in/on videos of bad quality;

♦ It is observed that at 9x9 division schema, the result is the closest to this of (FAST) method, i.e. the approach of more detailed uniform division could reach the results with feature points like (FAST), which determines the trend of our future work.
Live demo is available
5. Conclusion

- An innovative, efficient and robust method for 2D video stabilization is presented here, which is designed for real-time working on portable devices.

- The BSC (Boundary Signal Computation) chip of TI (Texas Instruments) is emulated herein for searching of correlations between the 1D integral projections, horizontal and vertical ones by the SAD (Sums of Absolute Differences) approach.

- The proposed method is based on an accurate vector model of the motion, allowing interpretations of increasing complexity for the transformations among frames (e.g. considering of scaling, skew).

- The method is intended as an initializing tool for the system of inertial sensors of a mobile device, finally responsible for the actual video stabilization.
References

THANK YOU 😊