Real-time Structure from Motion for Augmented Reality

Mark Pupilli and Andrew Calway

Technical Report, Department of Computer Science, University of Bristol, October 2002

Abstract

Our work is focused on developing a real-time structure from motion (SfM) algorithm that is usable in an augmented reality system. We envisage augmented reality applications involving a head-mounted camera and display system. This requires an SfM algorithm that is robust to different scene structure and camera motion and will invariably have to deal with the problems of occlusion, clutter and multiple objects. We aim to develop a system that would be capable of providing camera ego-motion and scene structure in a variety of such head-mounted video sequences. This report reviews current approaches to SfM and describes our current work on a Monte Carlo approach to the structure from motion problem.

Contents

Figure 1: A head-mounted display

1 Augmented Reality

In the most general sense augmented reality could refer to any technology that modifies a user's perception of reality. For our purposes augmented reality means the addition of computer generated text and graphics to what a person might normally see. This would be achieved through wearing a head-mounted display and camera. The camera takes in images then processes and modifies them before passing the result to the display. This allows the addition of graphics to enhance the user's view. This could be useful in various applications. For example virtual tours where the user is directed or given extra information through the display. It could also be used to give a person remote directions or instructions by directly manipulating what they see.

When using a head-mounted display as an augmented reality device it is desirable that the augmentation (graphics/text) be integrated with the real-world in a natural way. For example if 3D graphics are to be overlayed the virtual objects should move as if they are part of the real world scene. To do this effectively we need to know how the head-mount is moving relative to the world. An instructor might want to be able to draw or write on a surface to give directions or to position a virtual object in the scene. To place virtual objects accurately we need to know about the scene structure, and likewise if we want virtual objects to be occluded by real objects. There is also an issue of how real the virtual objects look compared to the rest of the scene. However, we shall only concern ourselves with realism in terms of motion and position.

1.1 Research Overview

One way of providing the motion and scene structure data is by computer vision based Structure from Motion. That is, given a video sequence from a head-mounted system, track features or objects and calculate the most likely camera motion and scene structure based on these tracking observations.

Structure from Motion (SfM) has been the subject of research in computer vision for more than twenty years. While much progress has been made in theoretical aspects of the area engineering practical solutions is a less developed art [1]. The most concentrated effort seems to have been the development of *off-line* augmented reality for special effects in film and television. Real-time Structure from Motion has not received as much attention, particularly when it comes to dealing with tracking noise, occlusions and multiple moving bodies.

There have been attempts to apply real-time augmented reality to specific user tasks, such as architectural inspection [2]. However, we hope it will be possible to develop a system that can be configured for different tasks. Changeable parameters might be the type of motion that is typical to the task or kind of structural information that is required about the scene. For example, in a known environment the scene structure may be partly or wholly available *a priori*. A vision based approach should be more flexible in this regard. In various configurations there may also be data from accelerometers and GPS available and the benefits of fusing this data with vision based SfM estimates is open to consideration.

2 Real-time Structure from Motion

"any one SfM algorithm is unlikely to perform well in all situations" J. Oliensis [1].

As a consequence the most important part of our research is engineering an SfM solution that is tailored towards the demands of augmented reality applications. That is, SfM that can operate in real-time, with video from a head-mounted camera, requiring little user calibration, dealing with occlusions, clutter, and possibly tracking the motion and structure of multiple bodies. A solution that can deal fully with such issues is not trivial and current SfM solutions do not cater for these requirements satisfactorily.

2.1 Current Approaches to SfM

Our choices are dictated by the gearing towards real-time SfM. Batch approaches to SfM, besides being computationally expensive, do not use data sequentially so will not be considered in great detail [5].

So called fusion approaches update the SfM solution as new image data is obtained. A popular way to do this is with variants of the Kalman filter [3, 4]. Oliensis [1] describes the benefits and drawbacks of fusion versus batch approaches. He makes the point that a fusion solution is only as robust as the local updates made to the SfM solution while a batch solution can use the whole sequence of data to adjust the global solution: inevitably leading to better results.

To solve SfM one must observe the structure somehow. There are schemes that track higher level features such as lines and curves but for a real-time solution tracking complex features will not be efficient without specialised hardware. As a result we are limited to point based tracking. It is possible to constrain a set of points so that the solution is more stable, for example a set of points can be clustered and constrained to lie on a plane as in [6].

2.2 The Role of Noise

Even a good feature tracker will produce tracks that have a variance of about one pixel. Many algorithms use unrealistic assumptions about the nature of these measurements. Soatto and Brockett show in [7] that the presence of noise - even in the order of the best feature trackers - leads to ambiguous SfM solutions that are inherent and not an artifact of a particular algorithm (they are even present in our own visual system). Since SfM is an optimisation problem, these ambiguities manifest themselves as false minima in the structure from motion error surface [8]. A consequence of this is that many proposed algorithms fail to deal with a variety of video sequences and it is not always clear what kinds of sequences a particular algorithm is appropriate for [1].

In batch solutions robust statistical methods such as RANSAC can be used to identify outliers to some noise tolerance. However, such tolerances are necessarily higher than that at which ambiguities would be removed [7] and in removing outliers you are possibly throwing away information about the motion and structure. Forsyth et Al. develop a robust Monte Carlo scheme for dealing with outliers while still utilising as much information as is possible from all tracks [9].

In a Kalman filter approach noise tolerances are set by the input co-variances to the filter. To get a solution with such an approach these input parameters must

be hand tuned (as well as removing outliers beyond these tolerances) see section 3 for details. Note that the Kalman filter also assumes that measurement errors are Gaussian which may not be realistic in all cases.

One way of avoiding problems due to feature tracking errors is to perform SfM without feature tracking [10, 11]. In Dellaert et Al's approach the corresponding features are to be determined in conjunction with solving the structure and motion using an iterative EM scheme. Unfortunately the lengthy convergence of the iteration prohibits real-time application.

Roy and Cox on the other hand solve the ego-motion of the camera literally without any correspondence information at all (albeit without structure as a consequence). Unfortunately this algorithm is only useful in two-image cases, requiring some method to unify scale between multiple images. The two image solutions could provide local information for a fusion scheme however.

Promising developments have been made in the use of Monte Carlo statistics to solve SfM. Sequential importance sampling is used in [16] to keep track of the distribution of camera parameters and scene structure at each step in a fusion scheme. Such a scheme should completely capture the noise sensitivity of SfM in the posterior state distribution. We are currently experimenting with an approach along these lines (section 3).

It should be noted that there have been some attempts at real-time SfM, [14, 5] for example implement real-time feature tracking and motion estimation.

An in depth critique of current approaches to SfM is due to Oliensis [1].

2.3 Multi-Body SfM

Within the noisy feature tracking results lie two types of outliers. The first are gross tracking errors where the feature being tracked has been confused with a similar nearby feature or has been lost due to occlusion or leaving frame. The second are features that are inconsistent with the cameras ego-motion because they are located on independently moving objects.

To reject the later type as outliers is not wise as it was demonstrated in [12] that utilising such features can improve the camera ego-motion solution. Of course it might also be required to solve SfM for independently moving objects.

Intuitively, identifying multiple moving objects is related to identifying outliers: instead of discarding the outliers you would like to try and reclassify them as being consistent with a separate moving object.

In a robust scheme due to Fitzgibbon and Zisserman RANSAC is used with a batch optimisation scheme to discover the dominant motion. Points consistent

Figure 2: AR in maya: virtual cube added to video. Rendered off-line in Maya. Some examples are available on the web *http://www.cs.bris.ac.uk/˜pupilli/MoCap.html*

with this motion are then removed and the next most dominant motion is extracted. This seems to work well but at a high computational cost [12].

There is a recursive scheme in [13] where feature points are separated into clusters. These clusters are then analysed to see if they are consistent with a rigid moving object. If they are then some SfM regime is used to solve for its motion. Otherwise they are recycled and reclassified. Possibly, this approach is more suited to real-time application.

3 Towards Real-Time SfM

Our first approach was based on the Kalman filter algorithm of Azarbayejani and Pentland [3]. This algorithm can work exceptionally well given the right noise conditions and given that the filter is properly tuned [4]. Tuning the filter by hand however is inconvenient, particularly for the applications we have in mind.

3.1 The Kalman Filter for off-line SfM

Regardless, we have developed an off-line AR tool using the Kalman filter approach. We developed a motion capture tool for Maya [1] that, given a hand-held

[1] Maya is a trade mark of AliasWavefront. www.aliaswavefront.com

video sequence and a tuned filter, streams the camera motion and scene structure into the Maya application. Maya can then be used to view the scene and insert virtual objects to the video sequence (see figure 2). Since we uses motion capture it should be possible to produce real-time previews in Maya, given a real-time SfM system.

As mentioned the Kalman filter algorithm proved unsuitable for real-time application. Typically Kalman filters are used in engineering applications where noise characteristics of the measuring devices are well understood. In SfM we have no way of knowing the noise covariance *a priori*. A brief attempt was made at automatically tuning the filter but this proved difficult for general scenes and computationally expensive. The main issue is that the filter has to be given time to converge before a judgement can be made to the viability of the solution which is time consuming. Judging how good a solution is and how close one is to the best set of input co-variances is not trivial. Judging a good choice of parameters requires knowledge of the noise characteristics of the feature tracker and the camera motion which we have no way of knowing in general scenes.

More importantly the time complexity of the filter is cubic in the state size. As a result extending the algorithm to deal with multiple objects is expensive. Even dealing with adding or removing tracked features is not trivial implementation wise as it requires a variable filter state size. That and the difficulty of segmenting multiple motions recursively within the Kalman filter framework convinced us not to pursue this approach.

3.2 Conditional Density Propagation

The Structure from Motion approach we are currently investigating is based on the Condensation algorithm [17]. This algorithm was developed for visual tracking applications but there is no reason why it cannot be applied to general state tracking: in this case the state to be tracked contains the camera parameters (structure is assumed known at present).

The principle advantage of a sampling approach is that the full probability distribution of the camera motion is captured at each stage: the distribution is possibly multi-modal with peaks corresponding to the true motion and to various SfM ambiguities. It should be possible to identify which peak represents the true camera motion [16]. Another big advantage is that, unlike the Kalman filter approach, Condensation does not have numerous free parameters that require tuning.

At the moment Condensation is just being used to track the ego-motion of the

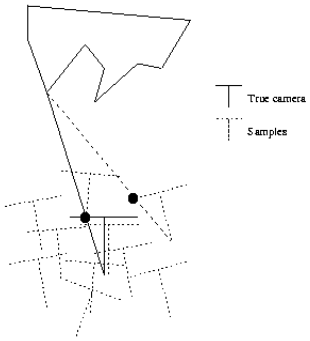

Figure 3: Illustration of sampling approach. Samples are are shown around the true camera position. By comparing how the scene projects into the samples to the true observations we can try to estimate the true state

camera. Samples contain the camera translation and rotation parameters (see figure 3). At each step samples are generated of possible camera motions. Using the known scene structure, projections into each 'camera sample' are compared to the tracked feature positions and weighted accordingly. While this is for development purposes one can envisage applications where scene structure is known or partially known [15]. The camera geometry is identical to that used in Azerbayejani and Pentland's algorithm (see appendix A). The Condensation sampling strategy is detailed in appendix B.

Using this naive resampling strategy leads to difficulties that relate to the complex error surface of SfM. The main issue seems to be that at each time step small rotations around the camera y-axis can be interchanged with small translations along the camera x-axis. Similarly rotations around the x-axis can be interchanged with y translations. This phenomenon is well noted in the SfM literature, and detailed in [7].

Figure 4 shows some samples in the x-translation, y-rotation plane. It is clear that there is a ridge in the probability density. The direction of the ridge is constant through-out the sequence and trails the true state at each time step. Correlation with the direction of state transition appears to be coincidental. Unfortunately the ridge's extent is a large percentage of the 180 frame sequence so it is difficult for the Condensation filter to distinguish the true state. The ground truth for this video sequence was obtained with the Kalman filter algorithm.

Figure 5 shows the tx, ry probability densities for a synthetic sequence with

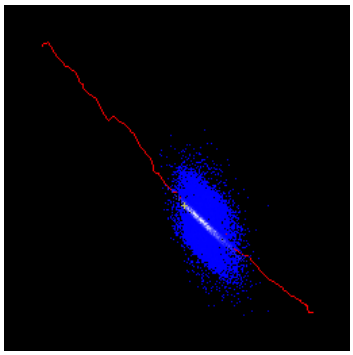

Figure 4: Probability density in tx, ry plane from one frame the pillar video sequence (below). Samples are blue/white: whiter samples have higher probability. The state-path is shown as a red line, the current state as a cross-hair.

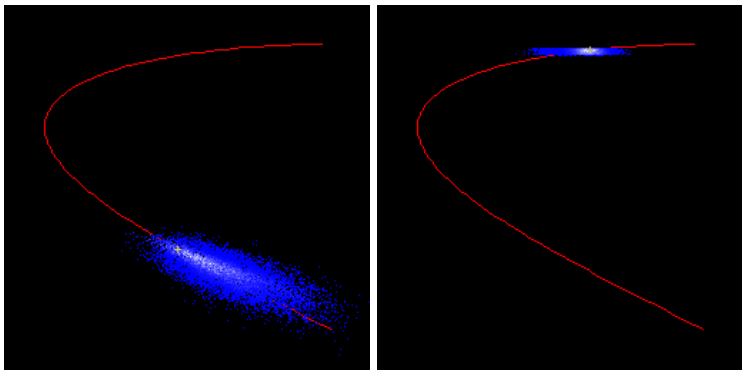

Figure 5: Probability Density in two frames from a synthetic sequence. In the later frame on the right Condensation has produced a tight distribution of high probabilities around the true state position

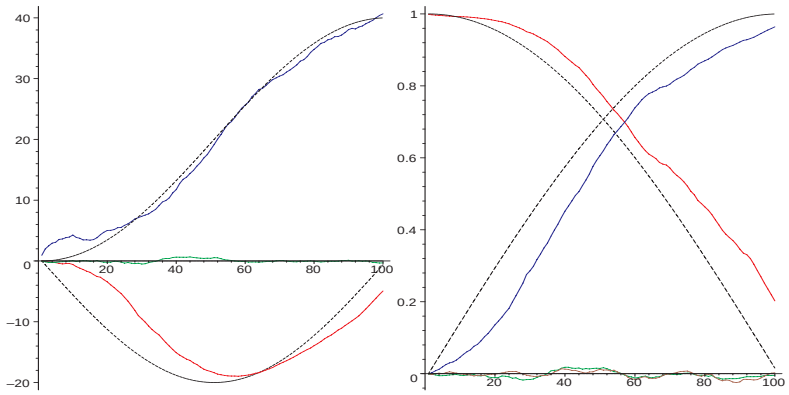

Figure 6: The left plot shows the 3 translation parameters and the right shows the 4 quaternion components. The ground truth is shown as dashed black lines

a different type of motion (half-circling a sphere, as opposed to moving towards and to the side in the pillar case). Again there is a ridge which maintains constant direction. In the first frame the ridge trails the true state because its direction coincides with the direction of motion. However towards the end of the sequence the ridge is more perpendicular to the state path resulting in a tight and accurate distribution. Running Condensation on this synthetic case shows acceptable results. Adding Gaussian noise of up to 3 pixels standard deviation has little effect on the results. The mean motion estimates are shown in figure 6.

3.3 Conclusions and Future Work

The results achieved so far are encouraging but not ideal. Because at some scale rotations can be interchanged with translations, Condensation seems to prefer to keep samples alive in their current position. Samples recieve higher weights in their current state because instead of translating a small amount then rotating a small amount they can do niether with indistinguishable errors. Hence, the distribution trails the true state, only moving on when the true state gets so far away that the samples have no choice but to move, or the motion becomes such that the range of ambiguity is shortened as in the later frame of figure 5. In many cases by the time the ambiguity has gone the sample distribution is too far from the true state to recapture the true distribution.

The main problem seems to be that the sampling strategy is too naive. Condensation can be dramatically improved by, and in most cases relies on, some

form of importance sampling. That is knowledge about the likely states of the system at the next time step. We hope to improve results by using more intelligent prediction in the resampling.

This leads us to point out a relevant difference in the Kalman filter approach. In the Kalman filter the measurements at the current stage are used to guide the solution but in our current Condensation method they are not used to guide the sampling. At the moment it is not clear how best to incorporate this extra information into the sampling. One would normally use a Kalman filter for such problems but this is impractical for the same reasons as using a complete Kalman Filter scheme.

Assuming that these problems can be solved we hope to extend the method to deal with unknown structure. It should also be possible to deal with multiple-bodies in a natural way since Condensation can be used to track multiple hypothesis: it should be able to keep track of multiple motions.

We would also like to incorporate learned statistics about human-head motion into the Condensation algorithm's prediction stage: this would better direct the importance sampling when focusing on augmented reality applications. If the user is performing a specific task then head-motion for that task may be quite constrained to certain movements. For example walking constrains head motion quite considerably in the direction of motion.

4 Acknowledgements

We would like to thank David Tweed for lending us his expertise. This work was funded through the Equator IRC www.equator.ac.uk.

References

[1] **John Oliensis** A Critique of Structure-from-Motion Algorithms. *NEC Research Institute.*

[2] **A. Webster, S. Feiner, B. MacIntyre, W. Massie, and T. Krueger.** Augmented reality in architectural construction, inspection and renovation. *Proc. ASCE Third Congress on Computing in Civil Engineering, Anaheim, CA, June 17-19, 1996, 913-919.*

[3] **Ali Azarbayejani and Alec P. Pentland** Recursive Estimation of Motion and Structure, and Focal Length. *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 17, No. 6, June 1995.*

[4] **Annie Yao and Andrew Calway.** Uncalibrated Narrow Baseline Augmented Reality. *In 1st International Symposium on 3D Data Processing Visualization and Transmission, pages 182–185. IEEE Computer Society, June 2002.*

[5] **Allessandro Chiuso et Al.** 3D Motion and Structure from 2D Motion Causally Integrated Over Time *IEEE Trans. on Pattern Anal. Mach. Intell, 2002.*

[6] **J. Alon and S. Sclaroff.** Recursive Estimation of Motion and Planar Structure. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition, (2000).*

[7] **Stefano Soatto and Roger Brockett** Optimal and Sub-optimal Structure from Motion. *In Proceedings of IEEE International Conference on Computer Vision, 1997.*

[8] **Jonh Oliensis** The Error Surface for Structure from Motion *NEC Research institute.*

[9] **D.A. Forsyth et Al.** Bayesian Structure from Motion. *Int. Conf. on Computer Vision (ICCV), pages 660-665, 1999.*

[10] **Frank Dellaert et Al.** Structure from Motion without Correspondence. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'00), June, 2000.*

[11] **Sebastien Roy and Ingemar J. Cox** Motion Without Structure. *NEC Research Institute.*

[12] **Andrew W. Fitzgibbon and Andrew Zisserman** Multibody Structure and Motion: 3-D Reconstruction of Independently Moving Objects. *6th European Conference on Computer Vision, June 26 - July 1, 2000, Proceedings, Part I*

[13] **Stefano Soatto and Pietro Perona** Three Dimensional Transparent Structure and Motion Segmentation and Multiple 3D Motion Estimation from Monocular Perspective Image Sequences. *Technical Report, California Institute of Technology, November 15th 1993*

[14] **Xiaolin Feng and Pietro Perona** Real-time Motion Detection System and Scene Segmentation. *Technical Report, California Institute of Technology, March 11th 1998*

[15] **Dieter Koller et Al.** Real-time Vision-Based Camera Tracking for Augmented Reality Applications. *In Proceedings of the Symposium on Virtual reality Software and Technology, 1997, pages 87-94*

[16] **Gang Qian and Rama Chellappa** Structure from Motion Using Sequential Monte Carlo Methods. *Proceedings of the Eighth International Conference On Computer Vision (ICCV-01), 2001, Volume 2*

[17] **Michael Isard and Andrew Blake** Condensation - Conditional density propagation for visual tracking. *International Journal of Computer Vision 29(1), 5-28.*

A Camera Geometry

For consistency with previous work the projection model and camera state samples are based on Azerbayejani and Pentland's formulation.

The mapping from world co-ordinates to image plane co-ordinates (projection model) is defined as follows:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} X_c \\ Y_c \end{pmatrix} \frac{1}{1 + Z_C \beta} \tag{1}$$

where $P_c = (X_c, Y_c, Z_c)$ is the 3D co-ordinate of a point with the origin at the centre of the camera's image plane. Here, β is the inverse focal length. That is for a focal length f:

$$\beta = \frac{1}{f}$$

The motion of the camera is described by a 3-vector translation (tx, ty, tz) and a 4-vector quaternion (S, qx, qy, qz). The quaternion is normalised as a 4-vector so only has 3 degrees of freedom. The translation is the absolute translation from the first/reference frame and the quaternion encodes the absolute rotation from the reference frame. In the original formulation tz was encoded as $tz\beta$. Since currently we are not estimating β we resorted to the basic version. Obviously, in the reference frame:

$$\begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} and, \begin{pmatrix} S \\ q_x \\ q_y \\ q_z \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

A 3D feature is located by its projection in the reference frame and its depth in that frame:

$$P = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} u \\ v \\ 0 \end{pmatrix} + \alpha \begin{pmatrix} u\beta \\ v\beta \\ 1 \end{pmatrix} \tag{2}$$

for known depth α and inverse focal length β.

B Condensation Formulation

Given the camera and projection model in appendix A we now describe the formulation for Condensation based SfM. That is simply, how to generate new samples from old samples and how to generate a fitness weight for each sample.

A sample only contains camera motion parameters since, for now, we are fixing the focal length to the true value and assuming the structure is known: for each feature we know its depth (and projection) in the reference frame. A sample is then a 7-vector:

$$(t_x, t_y, t_z, S, q_x, q_y, q_z)$$

with six degrees of freedom.

Given an old sample p_{t-1} we generate a new sample p_t as follows as follows:

1. Generate a uniform random 3-vector:

$$(\delta t_x, \delta t_y, \delta t_z) \sim U((-\frac{T_{var}}{4}, -\frac{T_{var}}{4}, -T_{var}), (\frac{T_{var}}{4}, \frac{T_{var}}{4}, T_{var}))$$

2. Generate a uniform random 3-vector:

$$(\omega_x, \omega_y, \omega_z) \sim U((-R_{var}, -R_{var}, -R_{var}), (R_{var}, R_{var}, R_{var}))$$

   rejecting samples with

$$|(\omega_x, \omega_y, \omega_z)| >= 4$$

3. Generate a new quaternion

$$\delta q = (\sqrt{1-\epsilon}, \omega_x/2, \omega_y/2, \omega_z/2)$$

where

$$\epsilon = \frac{1}{4}(\omega_x^2 + \omega_y^2 + \omega_z^2)$$

4. Generate p_t as:

$$\begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}_{p_t} = \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}_{p_{t-1}} + \begin{pmatrix} \delta t_x \\ \delta t_y \\ \delta t_z \end{pmatrix}$$

and using quaternion multiplication:

$$\begin{pmatrix} S \\ q_x \\ q_y \\ q_z \end{pmatrix}_{p_t} = \begin{pmatrix} S \\ q_x \\ q_y \\ q_z \end{pmatrix}_{p_{t-1}} \times \delta q$$

Here, T_{var} and R_{var} are tunable parameters that control the range of sampling for incremental translation and rotation respectively. NB: do not use system *rand()* since it gives results that are irrobust to the number of samples used.

The fitness of each sample is measured as follows:

1. Given a sample $p = (t_x, t_y, t_z, S, q_x, q_y, q_z)$ transform the n 3D features into the camera's frame of reference. Referring to equations 1 and 2:

$$P_c^i = R \times P^i + T$$

where T is the sample translation and R is the matrix equivalent of the sample quaternion.

2. Project the points P_c^i using equation 1 and calculate the Euclidean distances e_i in pixel co-ordinates to the location returned by the feature tracker. (Image-plane co-ordinates are converted to pixel co-ordinates by an arbitrary scale factor).

3. Calculate the goodness of fit of the sample as:

$$\frac{1}{n}\sum_{i=1}^{n}\cos^{16}(\frac{\pi e_i}{2E_{var}})$$

or 0 if $\frac{e_i}{E_{var}} > 1$

This measure is in the range $(0..1)$. E_{var} is a tunable parameter that controls the error cut-off in pixels.