

# Research Log

JeffGWood@mavs.uta.edu

June 4, 2016

---

March 30, 2016	Established research log after 3 hours of learning new $\text{\LaTeX}$
----------------	--

---

April 2, 2016	Added some additional comments to the <b>Process</b>
---------------	--

---

April 3, 2016	<p>Have been reading [ImageBasedRendering] [1].</p> <p><b>Question for Kamangar:</b> regarding [ImageBasedRendering] [1] about difference between:</p> <ul style="list-style-type: none"><li>• <b>Camera Plane</b> : Coordinates <math>u, v</math></li><li>• <b>Focal Plane</b> : Coordinates <math>s, t</math></li></ul>
---------------	---

---

April 11, 2016	<p>Reviewing blog articles located at:</p> <ul style="list-style-type: none"><li>• <a href="https://erget.wordpress.com/2014/02/01/calibrating-a-stereo-camera-with-opencv/">https://erget.wordpress.com/2014/02/01/calibrating-a-stereo-camera-with-opencv/</a></li><li>• <a href="https://erget.wordpress.com/2014/02/28/calibrating-a-stereo-pair-with-python/">https://erget.wordpress.com/2014/02/28/calibrating-a-stereo-pair-with-python/</a></li><li>• <a href="https://erget.wordpress.com/2014/03/13/building-an-interactive-gui-with-opencv/">https://erget.wordpress.com/2014/03/13/building-an-interactive-gui-with-opencv/</a></li><li>• <a href="https://erget.wordpress.com/2014/04/27/producing-3d-point-clouds-with-a-stereo-camera-in-opencv/">https://erget.wordpress.com/2014/04/27/producing-3d-point-clouds-with-a-stereo-camera-in-opencv/</a></li></ul> <p>for process to get webcam up and running. Previous issues related to fine-tuning <i>block matching</i> parameters. Need to review sources at list at bottom of <a href="http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html">http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html</a> to understand.</p>
----------------	---

---

April 19, 2016	<p>Made adjustments to python for image acquisition scripts (from blogs mentioned on April 11, 2016.)</p> <p><b>NOTE:</b> Consider creating rig with glue to keep stereo camera placement / direction constant.</p>
----------------	---

---

April 19, 2016	<p><b>UPDATE:</b> Error with <code>calibrate_cameras</code> python code causing linux machine to crash. If can't be resolved switch over to MacBook.</p> <p><b>NOTE:</b> Package should be setup by calling <code>\$ python setup.py install</code>.</p>
----------------	--

---

April 19, 2016	<p><b>UPDATE:</b> Crash due to recursive shell call and was fixed. OpenCV not detecting all chessboard corners. Will try a new board.</p>
----------------	---

---

April 20, 2016	<p>Did small amount of work on <b>Change of Reference</b> section in the paper. Added a section to the intro containing a map of commonly used symbols and notation.</p>
----------------	--

---

April 29, 2016	<p>Read following sections of [Chen1993] [2]:</p> <ul style="list-style-type: none"> <li>• Abstract</li> <li>• Introduction</li> <li>• Visibility Morphing</li> </ul> <p><b>SUMMARY:</b> Explicit Geometry is ignored (i.e. surface mesh and 3d-points). Geometry is kept in 2-d. Whereas Image Morphing interpolates between <i>pixel intensity values in fixed locations</i> the method in this article interpolates between <i>pixel locations with (relatively) fixed intensity values</i>. <b>Question:</b> Sections read mention that pixel positions are stored in 3d (3-tuple) data structure. I'm not sure I understand this correctly, since</p> <ol style="list-style-type: none"> <li>1. This would effectively make this structure a point cloud (but no mention of it in the paper).</li> <li>2. There is no mention of special "depth-based" hardware or cameras (Far as I know this is upposed to be a regular image).</li> </ol>
April 30, 2016	<p>Checked understanding of <i>epipolar constraint</i> through reading of [Hartley2004] [3] and its derivation of</p> $\begin{aligned} \mathbf{x}'^T \cdot \mathbf{E} \cdot \mathbf{x} &= \mathbf{x}'^T \cdot [\mathbf{t}]_{\times} \cdot \mathbf{R} \cdot \mathbf{x} \\ &= \mathbf{x}'^T \cdot \mathbf{l} \end{aligned}$ <p>and creation of MatLab code verifying this.</p> <p>I may have been mistaken about relation of <b>Fundamental Matrix</b> and <b>Essential Matrix</b>.</p> <p>My current understanding is the <i>Fundamental Matrix</i> describes point/epipolar line correspondance for images under <b>scale invariant</b> conditions (i.e. point correspondance and Fundamental matrix does not change when one image (or both images) are scaled (uniformly or omni-directionally).</p> <p><i>Essential Matrix</i> describes point/epipolar line correspondance for images under <b>normalized</b> conditions (i.e. unit-length is set equal to focal-length, and projection center is set at (0,0,1).</p>
May 2, 2016	<p>Additional wording to Stereo-vision section. I am unsure of best order to present ideas related to <i>multi-view</i> geometry.</p>
May 18, 2016	<p>Reviewed [Chen1993] [2] Section 2. Consider reviewing follow relevant articles:</p> <ul style="list-style-type: none"> <li>• Disparity [Gosh89]</li> <li>• Optical Flow [Nage86]</li> <li>• Look-up tables [Wolb89]</li> <li>• 3d scenes [Pogg91]</li> </ul> <p>Working on MatLab code to pick correspondig points in stereo-images, and calculate pixel offset vectors.</p>
May 19, 2016	<p>Read Section 2.3 of [Chen1993] [2]. View interpolation is limited by:</p> <ul style="list-style-type: none"> <li>• <b>Penumbra:</b> pixels visible in one source image <i>but not both</i></li> <li>• <b>Umbra,</b> pixels visible in neither source image, and <i>invisible</i> in destination image.</li> <li>• <b>Holes,</b> pixels visible in neither source image, but <i>visible</i> in destination image.</li> </ul> <p>Calculated formula for <i>pre-displaced</i> quad-pixel calculation using a bi-linear interpolation as:</p> $\mathbf{P}(u, v) = \mathbf{P}(0, 0) \cdot (1-u) \cdot (1-v) + \mathbf{P}(1, 0) \cdot u \cdot (1-v) + \mathbf{P}(0, 1) \cdot (1-u) \cdot v + \mathbf{P}(1, 1) \cdot u \cdot v$

May 20, 2016      Derived formula for  $uv$  calculation using *geometry matrix*, *blending matrix* and *basis vectors* of  $\mathbf{u} = [u \ 1]^T$  and  $\mathbf{v} = [v \ 1]^T$

$$\begin{aligned}x_{uv} &= [u \ 1] \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_{00} & x_{01} \\ x_{10} & x_{11} \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ 1 \end{bmatrix} \\y_{uv} &= [u \ 1] \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} y_{00} & y_{01} \\ y_{10} & y_{11} \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ 1 \end{bmatrix}\end{aligned}$$

**Question for Kamangar:** Is there a way given  $x$  and  $y$  to solve for  $u$  and  $v$ ?

---

May 22, 2016      Added more to thesis document.

Worked on singular-value of previous blending equation. where:

$$\begin{bmatrix} x_{uv} & 0 \\ 0 & y_{uv} \end{bmatrix} = \begin{bmatrix} \mathbf{u} & \mathbf{0} \\ \mathbf{0} & \mathbf{u} \end{bmatrix}^T \begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix}^T \begin{bmatrix} \mathbf{X} & \mathbf{0} \\ \mathbf{0} & \mathbf{Y} \end{bmatrix} \begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{v} & \mathbf{0} \\ \mathbf{0} & \mathbf{v} \end{bmatrix}$$

where

$$\mathbf{u} = \begin{bmatrix} u \\ 1 \end{bmatrix}, \mathbf{v} = \begin{bmatrix} v \\ 1 \end{bmatrix}, \mathbf{X} = \begin{bmatrix} x_{00} & x_{01} \\ x_{10} & x_{11} \end{bmatrix}, \mathbf{Y} = \begin{bmatrix} y_{00} & y_{01} \\ y_{10} & y_{11} \end{bmatrix}, \text{ and } \mathbf{M} = \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix}$$

---

May 23, 2016      Read [Chen1993] [2] section 2.4 on *Block Compression*.

**SUMMARY:** Blocks are established by *threshold* where each block contains pixels that are *offset by no more than the threshold*, allowing all pixels to be offset at once.

**Question for Kamangar:** Doesn't this assume that all pixels in the block have a uniform offset?

Working on MatLab program to perform pixel offsets of corresponding points (i.e. assign corresponding points to pixels in MatLab by non automatic methods)

---

May 24, 2016      Read following sections from [Chen1993] [2]:

- Implementations (3)
  - Preprocessing (3.1)
  - Interactive Interpolation (3.2)
  - Examples (3.3)
- Applications (4)
  - Virtual Reality (4.1)
  - Motion Blur (4.2)

**Question for Kamangar:** With regards to Section 3.1 and Section 1, why is a graph structure needed? Why is it a lattice?

**Question for Kamangar:** With regards to Section 4.1, I don't understand the concepts of *temporal anti-aliasing* and *super-sampling*?

Made additional changes / added material to thesis document.

---

May 25, 2016      Was using figures from <http://www.robots.ox.ac.uk/~vgg/hzbook/hzbook2/HZfigures.html> as test images, which may not be best source as there white borders, appear to be up-sampled, and do not contain (extrinsic) calibration info. Consider using images located at <http://vision.middlebury.edu/stereo/data/scenes2014/> that contain meta-info including (intrinsic) calibration info.

---

May 29, 2016	<p>Finished [Chen1993] [2]. Not sure if remaining article is of consequence.</p> <p>Finished MatLab program for <i>animating / hand-drawing</i> (See wording in [Chen1993] [2]) offset vectors. Program performs offsets in 2-dimensional space. Consider adding automatic <i>feature correspondance</i> and <i>z-buffer</i> information from depth map images available on MiddleBury database.</p>
May 30, 2016	<p>Point-correspondances do not follow even pattern as indicated in [Chen1993] [2]: <i>Bi-linear coordinates</i> and <i>quad partitionions</i>; May be better to use <i>Barycentric coordinates triangle partitions</i>.</p> <p>Read on MatLab <code>tform</code>, <code>maketform</code>, and <i>Delaunay</i> triangles for purpose of image partitions.</p>
June 1, 2016	<p>Read and finished [Park2003] [4].</p> <p><b>SUMMARY:</b> Multiple sections including <i>point correspondance</i> and <i>interpolation</i>. <b>Point correspondance:</b> Breaks images into rectangular partitions. Gets maximum horizontal and vertical pixel gradients using <i>Sobel operator</i> in each partition. The maximum gradient in each partition is thresholded to disregard homogeneous and textured regions. <b>Interpolation:</b> The images are partitioned with <i>Delaunay triangulation</i> using the point correspondances as triangle vertices.</p> <p><b>Question for Kamangar:</b> Article published seems to be vastly different depending on source (See <code>Park2003</code> folder). ScienceDirect version has more math and detail (maybe too much since it details what a <i>Sobel filter</i> is). Why would critical information, including algorithm steps and details, be omitted?</p>
June 2, 2016	<p>Reviewing PDF at <a href="https://staff.fnwi.uva.nl/l.dorst/hz/chap11_13.pdf">https://staff.fnwi.uva.nl/l.dorst/hz/chap11_13.pdf</a> for information on <i>tri-focal tensor</i>. Don't understand <i>practical</i> calculation of <i>fundamental matrix</i> from <i>Singular Value Decomposition</i> and <i>Linear Least Squares</i> (i.e. don't understand LLS calculation from SVD).</p>
June 3, 2016	<p>Working on implementing <i>triangle patch transform</i> in MatLab (using previously mentioned <code>delaunay</code>, <code>tform</code>, and <code>maketform</code> functions) needed for [Chen1993] [2] and [Park2003] [4].</p>
June 4, 2016	<p>Continuting work on getting triangular patches transformed in MatLab. Will use <code>affine2d</code> and <code>imwarp</code> instead of <code>maketform</code> and <code>imtransform</code>.</p> <p>Spent several hours on a false start trying to implement line drawing on pixel data, in order to implment polygon seperation. Finally found MatLab's <code>roipoly</code> function which does what I need.</p>

## References

- [1] Sing Bing Kang Heung-Yeung Shum, Shing-Chow Chan. *Image Based Rendering*. Springer Publishing, 1 edition, 2007. Available online at: <http://link.springer.com/content/pdf/10.1007%2F978-0-387-32668-9.pdf> Pages cited are **Book Page** Numbers. Formula for **PDF Page** Number is (**PDF Page Number** = **Book Page Number** + 17).
- [2] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 279–288, New York, NY, USA, 1993. ACM.
- [3] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.
- [4] Joon Hong Park and HyunWook Park. Fast view interpolation of stereo images using image gradient and disparity triangulation. In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, volume 1, pages I–381–4 vol.1, Sept 2003.