An Introduction to Stereo Vision and Disparity Computation

Edwin Olson, eolson@mit.edu
Melissa Hao, mhao@mit.edu

Introduction

Depth perception is an increasingly important subject with applications ranging from autonomous planetary rovers to automatic 3D object capture. While humans tend to take depth perception for granted, judging depth is difficult for computers, and remains a subject of ongoing research.

The fundamental process involves finding corresponding points in two different views of the same scene. Simple triangulation can then be used to determine the distance.

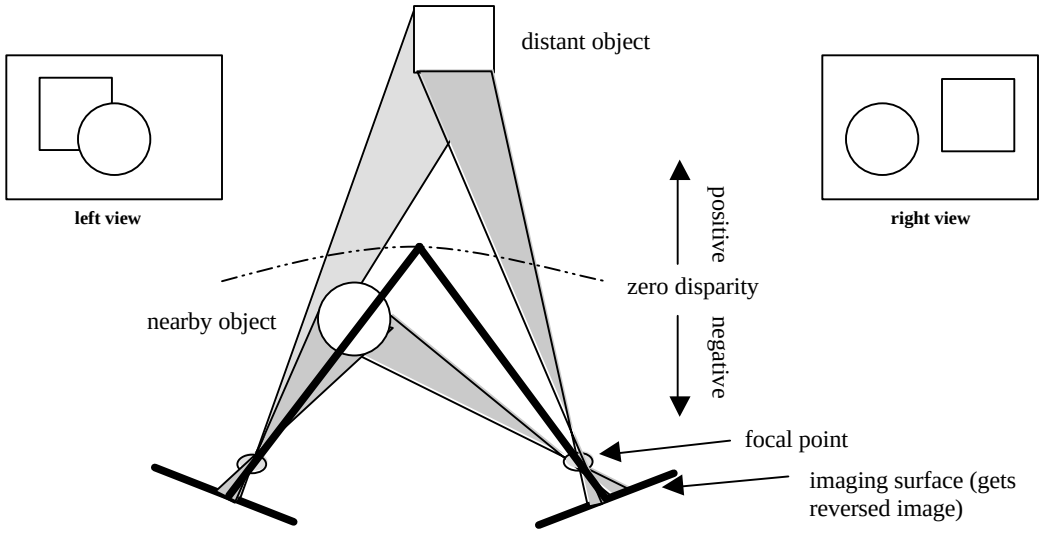


Figure 1. Geometry of stereo vision for one camera configuration.

Figure 1 depicts a typical camera configuration, with the cameras pointing somewhat inwards. Suppose the coordinates of two corresponding points are (x_{left},y_{left}) and (x_{right},y_{right}). For cameras that are properly aligned, $y_{left}=y_{right}$. The *disparity* is defined to be $x_{right}-x_{left}$. This value can be positive or negative, depending on the angle of the cameras as well as the distance to the object.

Searching for corresponding points is a recurring problem in machine vision as well as in image and video compression. Computing stereo disparity is very similar to finding optimal motion vectors, and approaches for both problems are similar.

Block Matching Algorithm

The most common approach in both stereo disparity calculations and motion compensation is to slide a block taken from one image over a second image. This approach is known as the *Block Matching Algorithm.* At each possible offset, a square-sense error is computed. Finding the position where the sub-images are most similar (and the minimum error occurs) is equivalent to computing the disparity.

Disparities typically have a small dynamic range (often < 10 pixels) compared to the actual distances to objects. Therefore, measuring disparities to integral pixel values results in very low depth resolution. The

solution is to measure disparities to subpixel resolution, with half-pel accuracy being common and quarter-pel used in some systems.

Finding corresponding points for every pixel in an image is an extremely computationally expensive task. Consider a straightforward implementation: for every pixel in the left image, a surrounding block of pixels (often 16x16 or 32x32) is slid across a row from the right image (which is the same height as the block from the left, but the width of the whole image.) At each position, the square-sense error (or other error metric) is computed, involving a large number of additions and multiplications.

Various optimizations intended to reduce the amount of computation have been proposed. Rather than searching an entire row, a subset of it is usually selected based on an estimate of the maximum disparity likely to be seen in the data. The search range can also be dynamically adjusted by exploiting the fact that nearby points are likely to have similar disparities.

Another class of optimizations relies on the observation that the error function as a function of horizontal offset (from which disparity is determined) is typically quite smooth, with a single and dramatic minimum.

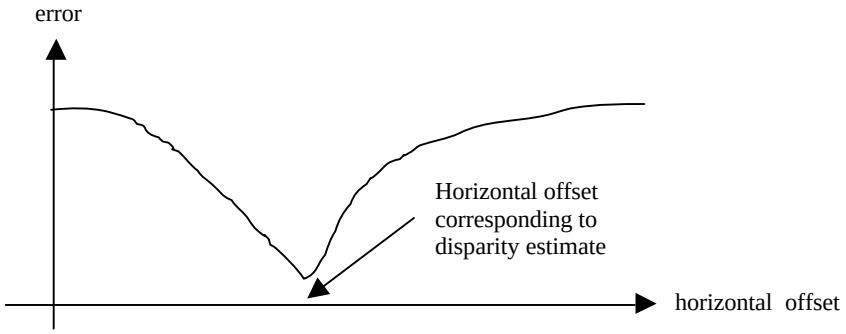


Figure 2. Typical error versus horizontal offset curve.

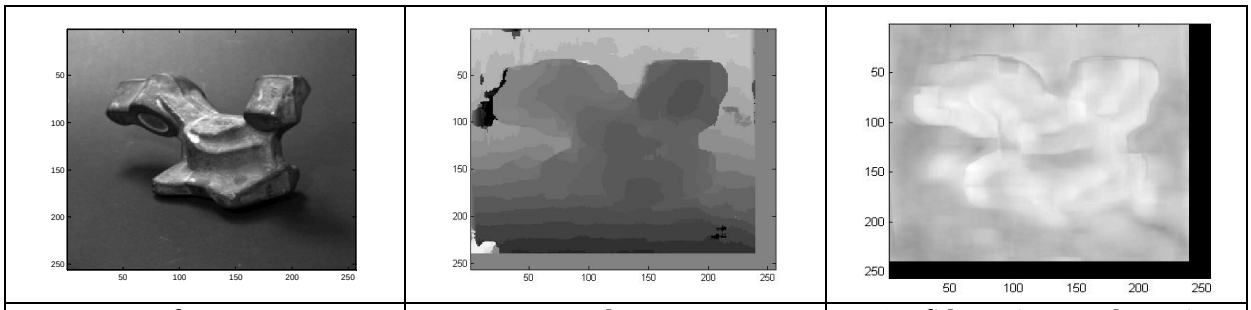
The smoothness of the error curve often makes it possible to find the minimum without an exhaustive search. For example, one can sample the error curve at a relatively small number of points, and select the best point(s) for further refinement. Logarithmic searches, common in motion compensation applications, employ this exact strategy.

Our Implementation

Using MatLab, we have implemented a stereo vision algorithm using a straight-forward block matching algorithm. We implemented half-pel accuracy using a ninth order FIR filter, rather than the lower-quality blinear filter often used. Performance was not a concern in our experiments. We typically used a block size of 8x8 or 16x16. Our program could also exploit color data, when available, to produce better error estimates. The error for a colored pixel was computed as the sum of the square of the difference in each color channel.

Areas in the test images with sharp edges and distinct features produced extremely sharp error curves, very similar to Figure 2, yielding excellent disparity estimation accuracy and consistency. However, areas of relative uniformity produced relatively flat error curves, resulting in highly erratic disparity estimates.

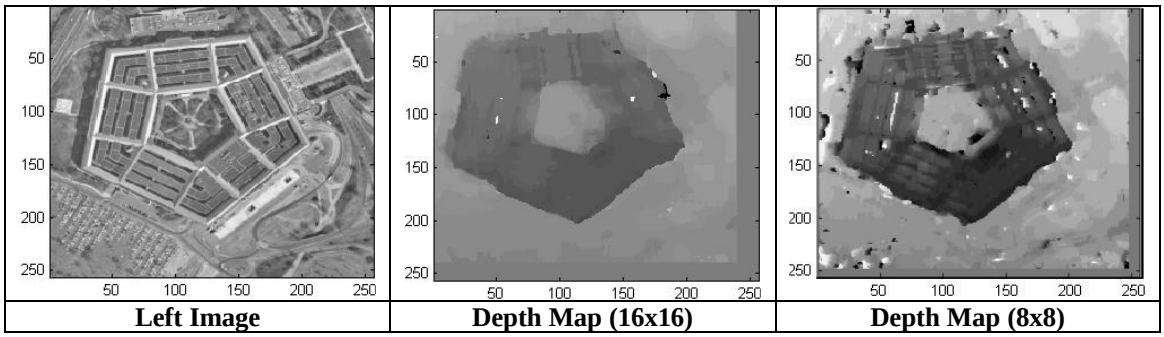
The sharpness of the error curve can be used to produce a confidence estimate. Error curves with a sharp and distinct minimum are typically very accurate whereas flat or erratic behaving error curves are less reliable. We experimented with several metrics to estimate confidence, including minimum error magnitude, minimum error scaled by average error, and interpreting the inverse error curve as a PDF and computing the expected error. Most of these simplistic approaches proved to work reasonably well, though they were typically conservative, reporting low confidences for data that was actually accurate.



Left Image	Depth Map	Confidence (expected error)

Figure 3. Half-pel depth map on Renault Automobile Part using 16x16 block matching.

Block size has a significant impact on the quality of the depth map. Larger blocks are more immune to noise and other differences (such as specular reflections and occlusions) that can occur between the left and right images. However, larger blocks cannot capture the fine depth detail that smaller blocks can. Figure 4 demonstrates a view of the Pentagon with large and small blocks. The Larger blocks provide cleaner data (fewer discontinuities), but many of the details are missing.



Left Image	Depth Map (16x16)	Depth Map (8x8)

Figure 4. Comparison of depth maps created with different block sizes.

Conclusion

Perceiving depth from multiple source images is a computationally expensive process. The block-matching algorithm used here has generally good performance, and the confidence estimate can be used to effectively mask out regions of high noise. Selection of block size is a trade-off between depth-map noise and detail. While only simple block-matching approaches were outlined here, depth perception remains a much researched topic.

MatLab source code and raw images can be obtained from: http://www.ravenousbirds.com/eolson/6.344