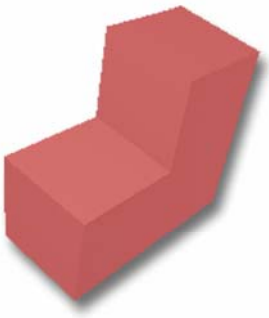


DEVELOPMENT OF AN EMBEDDED 3D ROBOT VISUAL SYSTEM

By Malcolm Stagg

1. ABSTRACT

Much work has been done over the years to find ways of generating a virtual image of an object in 3D. Most stereoscopic methods which currently exist, attempt to match every point from an image taken from a left image, to every point on an image taken on a right image. Not only is this very slow unless a great amount of computational power is used, but also the result is worthless when attempting to describe the object to a machine. At best, the distance to an obstacle can be calculated.

The methods I have been experimenting with are able to generate vectors of the left and right camera images to represent the 2D projection, and then attempt to reconstruct the 3D object by matching together the vectors. If this works correctly, it should allow the object to be shown in 3D space by a series of vector lines. These could be interpreted by a higher-level system to recognize the object, determine orientation and position of the object in 3D space, and make sense of simple scenes.

2. BACKGROUND INFORMATION

See "Robot Vision Algorithms" report.

3. PROBLEM

a. How can a microcontroller and computer based stereoscopic 3D visual system, for future use in a robotic device, be created to allow for the most accurate vector line modeling of a simple object in 3-dimensional space?

b. What positions – i.e. 3D translations and rotations – of a single-faced object will show up most accurately in 3D space, as well as preserve the area of the face?

4. HYPOTHESIS

a. If a visual system similar to the one described in the problem - part a is built, methods for edge detection, thinning, vectorizing, point-matching, and transforming to 3D space will probably need to be implemented, based on my prior research and previous project.

b. If testing of different transformations in 3D space for a simple single-faced object is completed, then the most accurate results for finding the transformation and area the face will be when the object is pointing towards the cameras, and is as close as possible to the them, because when less of the object is visible, or when there is greater distance to the object, small inaccuracies in measurement can lead to very poor results.

5. MATERIALS

- Computers for programming, viewing results, and debugging
- QBASIC language
- Visual Basic language
- C++ language
- MPASM microcontroller language
- Microcontroller programmer
- Microcontroller debugger
- Adjustable power supply
- 20MHz Oscilloscope
- 2 small grayscale video cameras
- 7 PIC18F452 microcontrollers
- 1 PIC16F628 microcontroller
- 2 LM1881 video sync separators
- 2 TDA8708 video analog to digital converters
- 2 MSM518221A video RAMs
- 4 74LS164 shift registers
- 2 DS1230-120 256k bit RAMs
- PAKII math co-processor
- 2 L293D stepper motor drivers
- 2 0.9 deg/step unipolar stepper motors
- Several connectors for debugger
- Several parallel ports & parallel headers
- 2 video camera power plugs
- 2 RCA plugs
- Several LM7805 & LM7812 voltage regulators
- 3 4049 Invertors
- 2 47ACT11008 AND gates
- Assorted resistors, capacitors, oscillators, inductors
- Breadboards
- PC Board etching supplies
- Drill press
- Soldering Iron
- Lots of 24 gauge wire
- 4 sockets for SOIC & SOJ ICs
- Jumpers & jumper headers

6. PROCEDURE

a. CONSTRUCTION

1. Research existing algorithms to include, and design new ones
2. Make demo software to test methods in Visual Basic
3. Create schematics for circuits
4. Build demo circuits on breadboards
5. Write programs in both Visual Basic to test and MPASM assembly to implement.
6. Use debugger to test programs – update circuits as needed
7. Test circuits with functional programs – use oscilloscope when needed
8. When a circuit works well, design a permanent PC board of the circuit
9. Etch printed circuit boards in acid (first of 7 boards completed)

b. TESTING

1. Cover base and create background of dark fabric for contrast
2. Cut a wooden triangle out of thin wood
3. Paint triangle white
4. Attach to dowel using a cork to allow for rotation and tilt
5. Place triangle in 5 positions, 5 rotations, and 5 tilts, while capturing images onto laptop
6. Find ideal vertices by hand for each image, also try edge detection and vectorization for one set of images
7. Create 3D models using computer calculations from angles, distances, and focal length of the cameras. Calculate average distance of the face and the orientation of the face as well. Test in 5 distances, 5 angles, and 5 tilts, record results.
8. Analyze data and make graphs.

7. RESULTS

These are the observations which show my accuracy of finding ideal vertices, as well as a test with the accuracy of the vectorization for placing correct vertices:

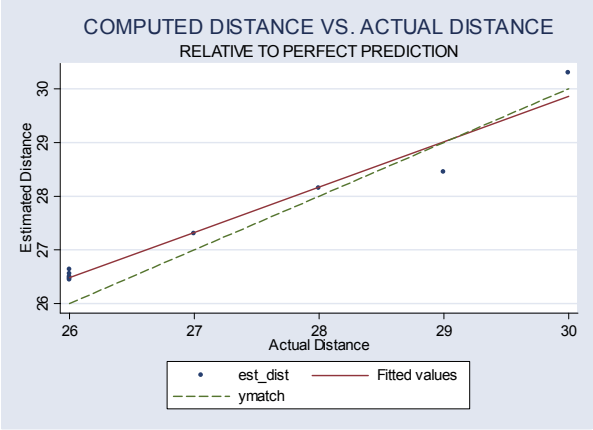
VERTEX PLACEMENT DATA												
Test	L 1 X	L 1 Y	L 2 X	L 2 Y	L 3 X	L 3 Y	R 1 X	R 1 Y	R 2 X	R 2 Y	R 3 X	R 3 Y
1	58	9	257	16	170	208	61	40	270	22	161	235
2	57	9	256	16	169	207	62	39	270	22	160	233
3	58	10	257	15	169	208	62	40	269	22	161	234
4	58	10	258	17	169	206	62	40	269	22	159	234
5	58	9	256	16	169	207	61	39	268	21	160	234
V	56	6	255	15	163	200	61	37	267	22	157	231

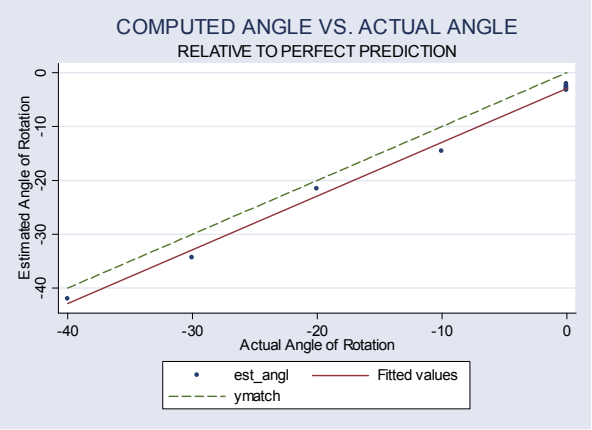
These observations show the measured distance, rotation, and tilt of the object vs. the actual:

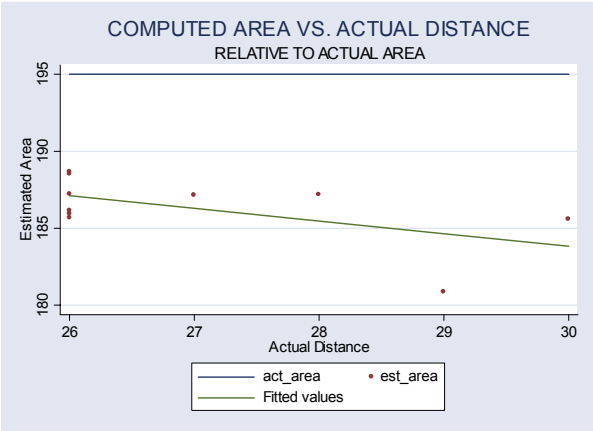
DISTANCE DATA WITH 0DEG ROTATION, 0DEG TILT, 0CM HORIZONTAL, 9CM VERTICAL					
Test	Rep	Actual Distance (cm)	Actual Area (cm square)	Est. Distance (cm)	Est. Area (cm square)
1	1	26	195	26.48	188.51
1	2	26	195	26.63	188.65
1	3	26	195	26.50	187.21
1	4	26	195	26.44	186.14
1	5	26	195	26.46	185.94
V	1	26	195	26.55	185.67
2	1	27	195	27.30	187.15
3	1	28	195	28.15	187.18
4	1	29	195	28.45	180.86
5	1	30	195	30.30	185.58

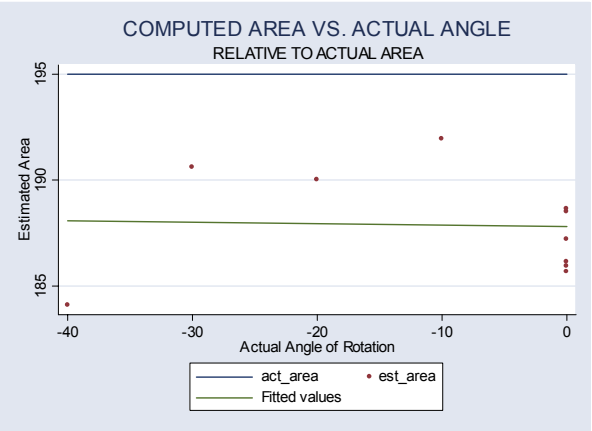
ROTATION DATA WITH 0DEG TILT, 26CM DISTANCE 0CM HORIZONTAL, 9CM VERTICAL					
Test	Rep	Actual Rotation (deg)	Actual Area (cm square)	Est. Rotation (deg)	Est. Area (cm square)
1	1	0	195	-3.25	188.51
1	2	0	195	-2.94	188.65
1	3	0	195	-2.53	187.21
1	4	0	195	-2.05	186.14
1	5	0	195	-3.10	185.94
V	1	0	195	-2.63	185.67
2	1	-10	195	-14.57	191.94
3	1	-20	195	-21.55	190.02
4	1	-30	195	-34.39	190.60
5	1	-40	195	-41.98	184.09

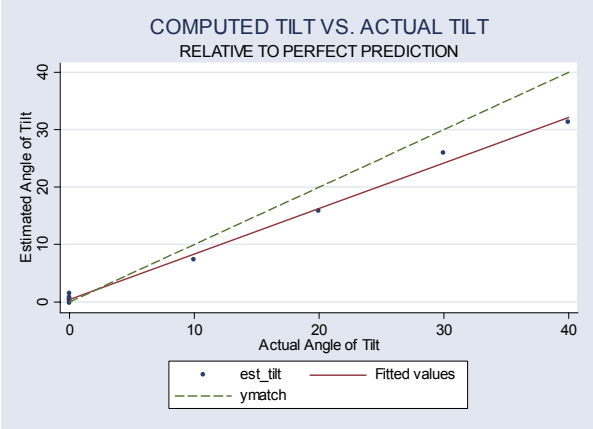
TILT DATA WITH 0DEG ROTATION, 26CM DISTANCE 0CM HORIZONTAL, 9CM VERTICAL					
Test	**Rep**	**Actual Tilt (deg)**	**Actual Area (cm square)**	**Est. Tilt (deg)**	**Est. Area (cm square)**
1	1	0	195	0.40	188.51
1	2	0	195	-0.23	188.65
1	3	0	195	0.76	187.21
1	4	0	195	0.22	186.14
1	5	0	195	0.51	185.94
V	1	0	195	1.44	185.67
2	1	10	195	7.31	189.74
3	1	20	195	15.84	189.12
4	1	30	195	25.94	184.74
5	1	40	195	31.31	183.64

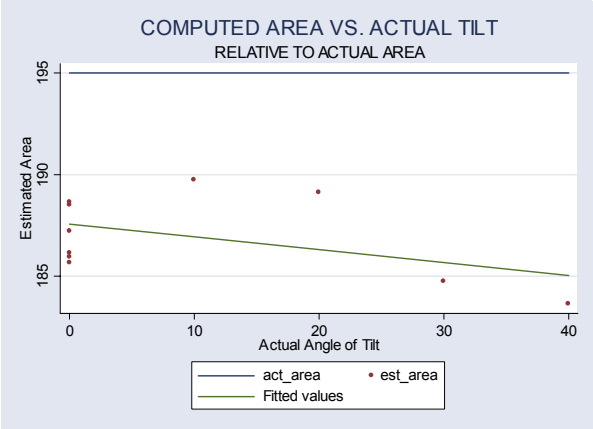












8. ANALYSIS

I did the analysis of all the data in the statistical program called Stata. This allowed me to make the graphs, as well as determine the significance of different factors using t-tests and regression analysis.

I started out with the graphs of computed vs. actual distance. These have computed distance as the y axis and actual distance as the x. There is a line showing the ideal values with a slope of 1. Linear least squares regression was done on the data to generate a straight line which represents it. With this data, the 95% confidence intervals are (.71 to .98). These values do not include 1 between them, so this means the calculated line is significantly different from the line of perfect prediction.

The plotted graph includes the ideal line with the slope of 1, the raw data points, and the regression line.

The next graph plots the area values at the different distances. Linear least squares regression is done on these data to once again obtain a straight line. The 95% confidence intervals for this line are (-1.8 to .17), which include 0, showing that this line does not change significantly with distance.

A plot was made with the actual area of 195 cm^2, the raw data points, and the regression line.

A t-test was done on these data to find if there was a significant difference between the calculated distance and actual. This returned the 95% confidence intervals which were (.08 to .56). These do not include 0, so this shows there was significant difference between the calculated and actual distances.

The significance between the distances obtained by hand-picking points and vectorization points was also looked at with a t-test. This gave the 95% confidence intervals of -.14 and .04. This shows the difference between distances from hand picked points and vectorization is not significant.

These same plots were repeated for rotation and tilt. Regression of the rotation data showed that the calculated slope was not significantly different from the ideal slope. The regression of the area values shows that area does not change significantly with the angle of rotation. The t-test for this data showed that the angle data was not calculated very well, and when the vectorization was compared to the other points with a t-test, it was found that they were not significantly different measurements.

For the tilt data, the regression line indicated that the slope is significantly different from the ideal value. The area regression showed that the area does not change significantly with tilt. In the t-tests, it was shown that the difference between calculated and actual tilt was not significant. Also, the angles obtained from the vectorization points were not significantly

different from the angles obtained by hand picking the points.

9. CONCLUSION

In conclusion to the experiment described in Problem – part b, my hypothesis was only partially correct. From these results, it does not seem that accuracy is improved when the object is close to the camera. It also shows that the distance is not calculated very well.

Distance vs. area data shows that the area calculations are always quite poor – approx. $10cm^2$ lower than the ideal values, and the results only slightly decrease in accuracy as the object is moved away. This is probably because of human error in the physical setup, or the sample object itself. Even if there is an error in one of these factors, the graph showed that distance has little effect on prediction of the area values.

The results for angle of rotation showed similar conclusions as the distance data. Accuracy of the calculated angle of rotation was very close to the ideal values, from the graph of the calculated vs. actual angle. However, the line representing the measured data is about 3 degrees below the actual rotation. This is likely also because of a human error.

The area taken after each of the rotations seems to have the same problem as the area data from the distance testing. It is still about 7 cm^2 below the ideal values, but angle of rotation has little effect on the accuracy.

However, when angle of tilt is examined, it can be seen that accuracy starts out very good, and gradually becomes less. This shows that the angle of rotation has an effect on the accuracy of this measurement.

The area data for angle of tilt shows the same problem mentioned above regarding the calculated are being less than the actual area, but also shows that angle of tilt has some effect on the accuracy of the calculated area of the object.

Overall, I am quite pleased with the accuracy of these calculations. They give calculated results which are quite close to the actual results.

The design of the system – see part a of the Problem – I was basically correct about the algorithms which would need to be found or designed to make it possible to generate the 3D data from the input camera views, except I did not think of some of the more generic algorithms which would be required.

For example, I did not anticipate in my hypothesis part a, that I would need filtering algorithms before edge detection, and non-maximal suppression instead of a thinning algorithm. Also, I did not think about the digital filters which are required before vectorization.

The distances and angles of rotation obtained from vectorization were not significantly different from those obtained from hand-picked points. However, the tilt data was much higher with vectorization placed points than with hand-picked points.

10. SOURCES OF ERROR

One main source of error is the alignment of the cameras. I found that one of the cameras had the CCD shifted down vertically by the equivalent of 28 pixels – a problem which was corrected using math – but this showed that the accuracy of the cameras may not be all that great.

Also, the orientation of the cameras may not be completely perfect. The vertical tilt of the cameras must be checked and corrected after each setup, to ensure the 8.4cm center of lens remains at 8.4cm when looking at a ruler at a distance from each camera. This may have drifted slightly while running the tests. A difference of a fraction of a millimeter in

rotation can create a huge problem in the calculations.

Another source of error may be the actual position and orientation of the object. The distances from the origin had to be measured in 3D space, so there may have been some small inaccuracies caused by poor measurement.

One of the sources of error from the analysis would probably be the number of data points without repetitions.

11. DISCUSSION

In the project this year, I have made significant progress from my previous project last year. Last year, I focused mainly on methods of edge detection and tried a little non-maximal suppression, but none of the microcontroller system worked last year, and the computer system was not nearly capable of making 3D calculations.

The problem for that project – Evaluation of 3D Object Recognition Methods – was:
"Which edge detection gradient operator will provide the most accurate results for correctly positioning lines, allowing corners to be detected properly, and giving the most correct results when 3D matching and calculation is completed? At which binary conversion threshold will it accomplish this with the greatest results?"
My objectives have changed quite a lot for this project.

Due to all my changing ideas for the vectorization algorithm, I unfortunately cannot currently demonstrate a fully functional version of that – either in-circuit or on the computer system.

Another problem which has been rather bothersome is the breadboards. These boards consistently have the problem of wires falling out, as well as shorts developing between close holes. With the analog video input, I also experienced problems with the capacitance to other signals decreasing the quality of the video image.

That is why I have been attempting to make Printed Circuit Boards of all the circuits. However, due to unfortunate printer problems destroying some of the boards, I have not had time to do any more than just the first board.

12. APPLICATIONS

There are a huge number of applications where a system like this could be used for, when it is fully optimized and constructed. One such application could be for the military. In the past, there have been some excellent missiles which are guided by 2D terrain images. If this was extended with a system such as this to 3D, it would have incredible capabilities.

Figure 1: This cruises missile uses 2D object recognition to find where it is located

The main application I was looking at during the construction of the electronics though was a basic robotic device. Such a device could be extended to be capable of finding an object and where it is in a scene, adding to a virtual model of an object, and recognizing what an object actually is.

A robotic device with those capabilities could certainly be of help to someone who is blind – to help them locate objects and warn of obstacles.

In the future, as well as very soon completing the algorithms and associated electronics, I

would like to extend this project possibly into the fields of motion and sound.

13. SPECIAL THANKS

1. **Dr. Peter Stagg**
   My uncle for his kind help and support especially with the 3D calculations and ideas for the experimentation.
2. **Vicki Stagg**
   My mother for all of her help, especially for her help with Stata for the analysis.
3. **Andrew Stagg**
   My brother, for all his computer related help and support.
4. **David Wells**
   Auton Engineering, Ltd.
   For his kind donation of a PicStart Plus PIC microcontroller programmer
5. **Damon Chu**
   Microchip Technology, Inc.
   For sending me a free MPLAB ICD 2 PIC microcontroller debugger
6. **Cory Mills**
   National Semiconductor, Inc.
   For sending me a free COP-8 microcontroller emulator
7. **Fanny's Fabrics**
   For the 40% discount on the black fabric

14. REFERENCES

[1] Brown, Christopher. Advances in Computer Vision Volume 1. Academic Press, 1989.

[2] Desai, Ujjaval et al. Edge and Mean Based Image Compression. http://www.ai.mit.edu/people/bkph/AIM/AIM-1584.pdf. 1996.

[3] Freeman, Herbert. Machine Vision for Three-Dimensional Scenes. Academic Press, 1989.

[4] Guzman, Adolpho. Computer Recognition of Three-Dimensional Objects in a Visual Scene. http://ncstrl.mit.edu/Dienst/UI/2.0/Composite/ncstrl.mit_lcs/MIT/LCS/TR-59/1?nsections=13. 1968.

[5] Iosoft Ltd. Chipvid Project. http://www.iosoft.co.uk/chipvid.php.

[6] Marr, David. Vision. W. H. Freeman and Company, 1982.

[7] Parker, James. Practical Computer Vision Using C.

[8] Roberts, L. G. Machine Perception of Three-Dimensional Scenes. MIT Press, 1980.

[9] Shirai, Yoshiaki. Three Dimensional Computer Vision. Springer-Verlog Berlin Heidelberg, 1987.

[10] Stagg, Malcolm. Evaluation of 3D Object Recognition Methods. 2003.

[11] University of Edinburgh. Image Processing Learning Resources. http://www.dai.ed.ac.uk/HIPR2/hipr_top.htm . 2002.

[12] Wahl, Freidrich. Digital Image Signal Processing. Artech House, Inc., 1987.