Real-Time 3D Object Tracking

R. S. Stephens

Computing Devices Co. Ltd,
Kings Drive,
Eastbourne,
East Sussex, BN21 2UE

A system for tracking the three dimensional motion of objects in real time is described.

The system uses a technique of localized image access that enables it to achieve frame-rate operation without the need for any special purpose hardware. All the processing for one camera can be performed on a network of five transputers, with a single transputer combining the information from several cameras.

A dynamic Hough transform is used to solve a set of constraints derived from image measurements. I shall demonstrate how the robustness and intrinsic parallelism of the Hough transform can be put to practical use.

The system has been tested on a stored sequence of one hundred image pairs, which were captured under realistic, noisy conditions, and the results are presented here.

The conventional wisdom in artificial intelligence, and particularly in computer vision, is that large volumes of input data should be converted into a more compact symbolic form by a process of segmentation. The difficulty of performing a satisfactory segmentation of image data (except under the most artificial of conditions) has caused the greatest problem in implementing useful vision systems. Not only are segmentation algorithms computationally expensive (which often prohibits real-time implementation), the errors that they introduce into the symbolic description of the image can be misleading to the higher level processes. This is basically why the robustness of many vision systems is so poor, and I was interested to see whether, by cutting out the segmentation phase, a truly robust vision system could be implemented.

The system described here uses a technique of direct access to the raw image data for measurement of the lateral displacements of edges from a predicted position. A similar technique was used by Brisdon [1] for verification of model instances in the Alvey Vehicle Exemplar. She found that it gave a more reliable indication of the presence of the vehicle than the data-driven approach based on segmentation.

The task performed by my system is that of tracking the three dimensional motion of a rigid body, so, if the system is to work in real time, speed is a major consideration. Previous work on unconstrained motion tracking (eg, [2,3]) involved segmenting the image, then matching features between consecutive frames to determine the motion. By avoiding the need for segmentation, I have have been able to implement the system entirely in software running on a small network of transputers. The system relies on the sampling rate being fast enough, in relation to the motion, to avoid problems in forming the right correspondence between image features in consecutive frames.

One application of a cheap, robust, fast object tracker is for the visual control of a robot. Currently the position of a robot's arm is deduced from joint position sensors, and is susceptible to a build-up of errors due to the fact that the arm is not perfectly rigid. Visual measurement of the position of the end-effector would be unaffected by the deflection of the arm, so the arm could be made of a lighter (more flexible) material. This would be highly desirable for the purposes of controlling the robot, and would allow much faster and better controlled movements to be performed. To control such a robot, however, a sampling rate well in excess of the standard 50 Hz frame rate would be required, and further development of the system will be necessary before this is possible.

DESCRIPTION OF THE SYSTEM

The system is given a geometric model of the object to be tracked, consisting of features that are likely to cause a discontinuity of grey level in the image. Currently the system only uses edges and surface markings as features, but cylindrical and spherical surfaces giving rise to "horizon lines" could also be modelled. Each feature consists of a point on the edge - the position of the point and the direction of the edge on which it lies are stored, using the object's coordinate frame. Each feature also has a coarse lookup table to determine whether or not it is visible from a given viewpoint. These features were input by hand in order to test the system, but they could be generated automatically or interactively with a CAD system.

For each new video frame the system is given an estimate of the position of the object, either from previous frames, or from knowledge of the initial position. Using knowledge of the camera's projection matrix, the system then predicts the location and orientation of a number of edges in the image, and processes the image in small regions to locate edges with the expected orientation close to the predicted positions. The lateral displacements of the edges from the predicted positions are measured to sub-pixel accuracy by smoothing the image data.

A sufficient number of these measurements enables the true position of the object to be determined. If the estimated position is close to the true position, the constraints imposed by the edge displacement measurements can be linearized. Although perspective effects do give a measure of range, the equations will not be well conditioned if just one camera is used. However, by combining measurements from two widely spaced cameras, a well formed set of equations is much more likely to be obtained.

AVC 1989 doi:10.5244/C.3.15

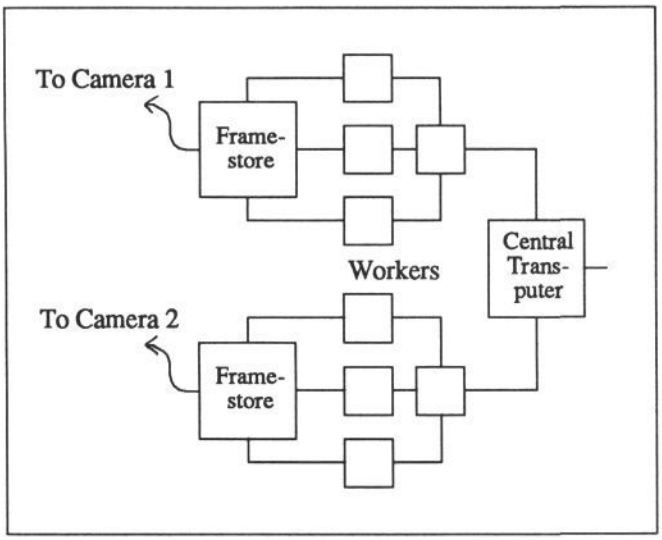

Figure 1. Minimum Transputer network for frame-rate operation

Because of adverse lighting or obscuration, some of the measurements will be incorrect. If no edge of the correct orientation can be found near the expected position, then no harm is done since the measurement is simply ignored. However, where an edge is found that is not caused by the feature in question, a spurious equation will be generated. If the system is to be robust, these spurious measurements must be excluded from the solution of the set of linear equations, so a simple least squares approach cannot be adopted. The method I have chosen to solve the equations involves the use of a Hough transform, and will be described later.

The system relies on the availability of a good estimate of the initial position of the object (within, say, 5mm), and no provision is made for recovery of the object in the event that it is lost. It is also assumes that the object moves in a reasonably smooth or predictable manner.

For testing purposes the system was run on a single T800 transputer, but for real time implementation the network shown in figure 1 would be used. This configuration would be sufficient to track an object under conditions of low noise, and where great accuracy is not required.

The network consists of two framestores, each of which captures and processes images from one of the cameras in a double-buffered mode. Each framestore is accessed by a T800, and all the edge displacement measurements are performed by these processors, so there is no need to communicate image data between processors.

The worker processors carry out two tasks: the projection of model edge points into the image, and the formation of the Hough transform. The Hough transform is parallelized by dividing the edge points up among the workers, so the major communication overhead in the system is that of summing the Hough spaces from all workers. The workers are configured to minimize the communication overhead. The worker processors consist of T800s, and the processes running in them are simple enough to fit entirely in the 4K of on-chip static RAM.

Finally, there is the central processor that combines the Hough spaces from the different cameras and interprets it to update the position of the object. Because the Hough transform is performed in global coordinates, the information from different cameras may be integrated simply by summing the Hough spaces. This is an example of the use of a Hough transform for the integration of information from multiple sensors - a use that has not been widely studied. Combining data from the two cameras at this late stage in the processing is more robust than combining them by stereo triangulation to determine the 3D displacement of edge points.

We shall now consider the two most important parts of the system, the measurement of edge displacements, and the Hough transform, in more detail.

EDGE DISPLACEMENTS

Given the estimated position of the object, the camera projection matrix, and the model of the object, the position and orientation in image coordinates of some likely grey-level discontinuities can be predicted. We are required to measure the lateral displacement of the nearest significant edge from the predicted edge.

Canny's technique [4] of using elongated Gaussian convolution kernels for detecting edges of a particular orientation is employed, together with his non-maximal suppression and sub-pixel location algorithms. Kernels of a fixed size are stored for a sequence of orientations at 5 degree intervals. The nearest kernel to the predicted edge orientation is selected, and is applied to the image

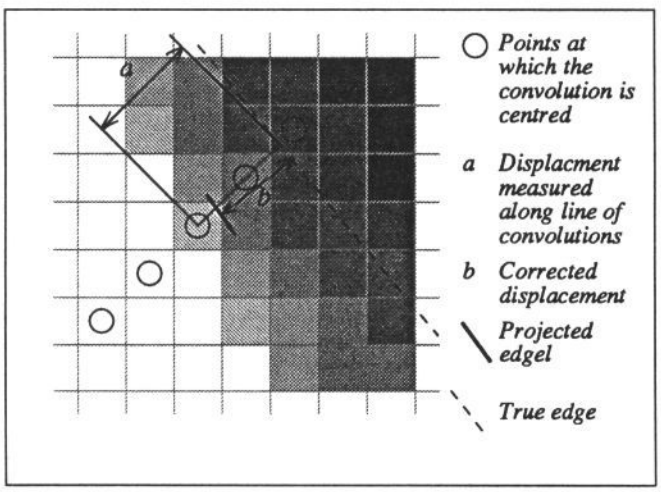

Figure 2. Measurement of Edge Displacements

sequentially along a line at the nearest 45 degree interval to the normal to the edge (see figure 2). Floating point maths is used for the convolution. As the kernel is moved outwards in both directions, the two sequences are differentiated. The process stops either when the gradient magnitude (at a point of maximal gradient) exceeds a threshold, or when the line has been extended more than a certain number of steps. Quadratic interpolation is used to locate the gradient maximum to sub-pixel accuracy. The resulting measurement is then corrected for the sub-pixel offset of the predicted position, and for the angle between the 45 degree interval and the predicted edge orientation to give the true edge displacement.

The processing described above might appear to be too

complex for a frame-rate implementation in software, but remember that it is applied only to a few small regions of the image. It is also worth pointing out that the T800 performs floating point maths very nearly as fast as it performs integer maths (which is very fast). The measured execution times on a 20 MHz T800 are shown in table 1. Frame rate operation is just possible using 3x3 convolution for about 20 edge points. Edge measurement could be speeded up by distributing the image data over two or more

	Edge displacement in pixels		
	0	2.5	5.0
9x9	1.66	2.90	4.26
5x5	0.63	1.10	1.64
3x3	0.49	0.61	0.79
no smoothing	0.19	0.22	0.27

Table 1. Time in ms for measurement of an edge displacement for various sizes of convolution kernel

processors, and splitting the edge points between processors, but a more cost-effective way would be to smooth the image in hardware before it is transferred into the framestore. This would allow edge measurements to be performed without smoothing, which, as table 1 shows, would be easily fast enough.

Figure 3 shows a typical set of measurements. Note that under normal circumstances, where the predicted position of the object is fairly accurate, most of the edge displacements are sub-pixel.

THE HOUGH TRANSFORM

The position of an object is described by six parameters: three for translation and three for rotation. These parameters span a six dimensional parameter space or Hough space, each point in the space representing a different position of the object in the real world. Because of its high dimensionality, it is impossible to hold the whole of this space as an array in a computer's memory, but a small region, centred on the estimated position, can be stored.

The use of a Hough transform to determine the six parameters of an object's position is described in [5], using vertex pairs instead of edge displacements as input features. The technique I am using has much in common with the "Fast Hough Transform" described by Li, Lavin and Le Master [6], although for use in image sequences I have decoupled changes in position from changes in scale.

Consider a six dimensional array of accumulators, 3 cells wide in each direction, all initialised to zero. For each edge displacement measurement, those cells that represent a position consistent with the observed measurement, are incremented (see appendix A for derivation of the test for consistency). The accumulator that has the greatest count, when all measurements have been considered, represents the most likely position of the object, and this is output as the current position of the object.

Hough transforms in general are robust, so spurious inputs

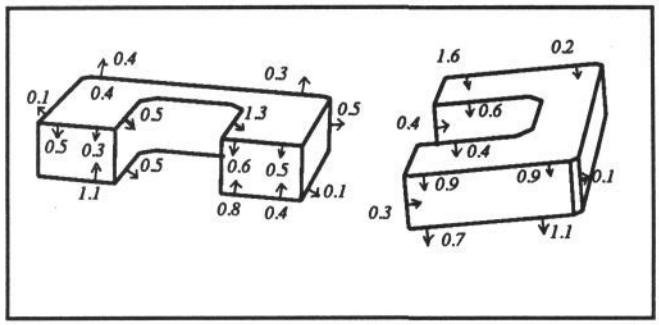

Figure 3. A typical set of edge displacement measurements from left and right cameras

do not affect the output in the same way that they would affect a least-squares solution. This is because it is the consistency of the data that causes a particular accumulator to have a large count, while inconsistent data will increment other cells in a purely random manner. This ability to sort the good edge displacements from the bad is essential to the robustness of the whole system.

Note that the output position is in effect quantized by the separation of the accumulator cells, so the use of the smallest possible cell size is desirable. If the cell were too small, however, the true position of the object might lie far outside the region of Hough space represented, and the accumulators would lose their significance. One way of increasing the accuracy of the system (at the expense of speed) is to perform multiple iterations of the Hough transform for each frame, with the cell size halved at each step. Although one can add more worker processors to the system, the extent to which one can do this will be limited by the communication overhead of combining the Hough spaces.

A process of coarse-to-fine control is used to adjust the scale of the Hough space according to the conditions. If the output position is close to the original estimate, the size of Hough space cells can be decreased, but if the estimate turnes out to be a poor one the cell size should be increased.

The arrangement of cells described above contains 729 (3 to the power of 6) cells. Although this would not be an impossibly large number of units to handle, an alternative arrangement, consisting of a set of 43 close-packed spheres, as shown in figure 4, was found to perform just as well. The close-packed arrangement is appropriate because the method of forming the Hough transform approximates the cells as spheres, and it is desirable to minimize the amount of overlap. Appendix B gives a method for generating a set of close-packed spheres in any number of dimensions (it is left as an exercise for the reader to prove that the method is valid).

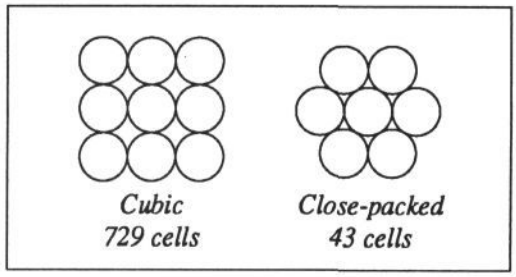

Figure 4. Arrangement of Hough space cells, simplified to 2 dimensions

Mapping	82 μs
Pre-computation	1.46 ms
Accumulation	0.24 ms

Table 2. Computation times per edge point for worker transputers

Table 2 shows times for computation of the Hough transform on a T800. The edge points are mapped into the image and are sent to the framestore processor. While the framestore is measuring the edge displacements, the worker processors pre-compute the allowable range of the displacement for each cell. Then, when the actual displacements are sent back from the framestore, the Hough space is accumulated. This use of parallelism means that neither the framestore nor the workers are idle for very long. The time taken to sum the Hough space across the network has been measured as 0.19 ms per layer of workers (the network in figure 1 has two layers of workers).

EXPERIMENTAL RESULTS

The system was tested using a sequence of 100 image pairs, which were stored on disk. Processing was performed on a single T800 transputer, using software written in OCCAM, making liberal use of communicating parallel modules. The sequence shows a robot moving a "widget" through a programmed set of points, pausing at each one while the images were captured and stored on disk. Use of the robot allowed the true position of the object at every frame to be known, so the accuracy of the system can be gauged objectively.

Figures 5 to 8 show a selection of images from the sequence, overlayed with the system output. To give an idea of the scale, the U-shaped object in the robot gripper is 70mm in length. The images are 256 by 256 pixels, but the object is only 50 pixels or so wide, so the data can be considered to be low resolution. A substantial part of the object is always obscured by the robot gripper, and the chequered cube (which was used for camera calibration) forms a confusing background, with many strong edges close to the object. This set of data was designed to test the robustness of the system to the limit, and there are parts of the sequence were the system does break down.

The output of the system over frames 10 to 80 is shown in figure 9, showing successful tracking up to frame 75, after which the object is lost. At several stages errors exceeding 10mm are generated, but the system is able to correct itself. A contribution to the final failure can be seen in figure 8 - edge points obscured by the robot gripper have become associated with the vertical edge of the gripper, and as the

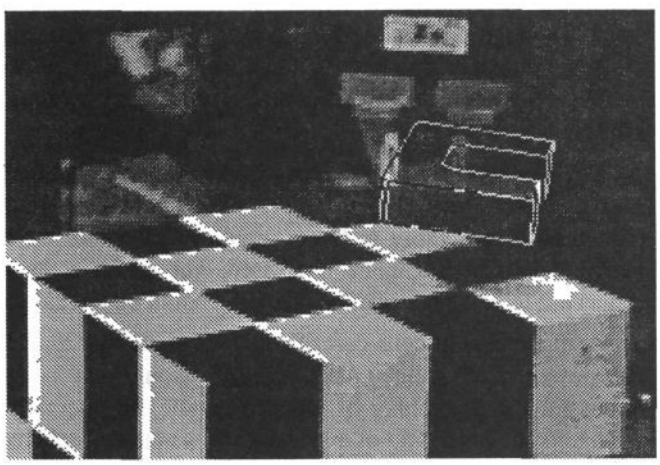

Figure 5. Left camera, frame 30

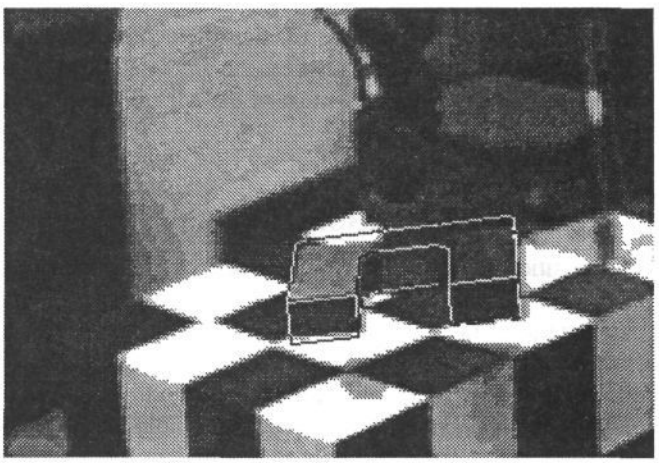

Figure 6. Right camera, frame 30

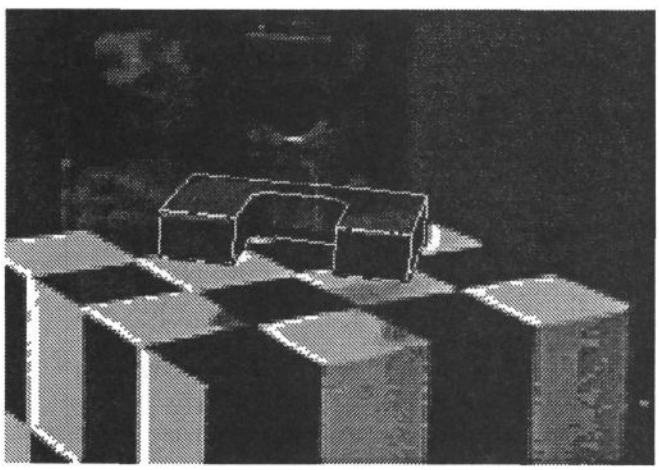

Figure 7. Left camera, frame 70

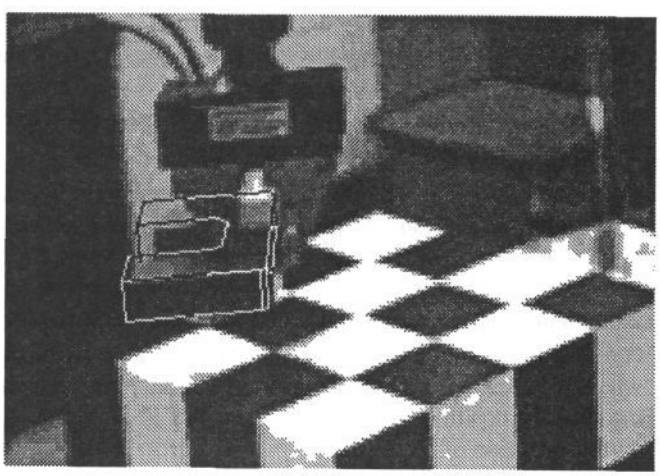

Figure 8. Right camera, frame 70

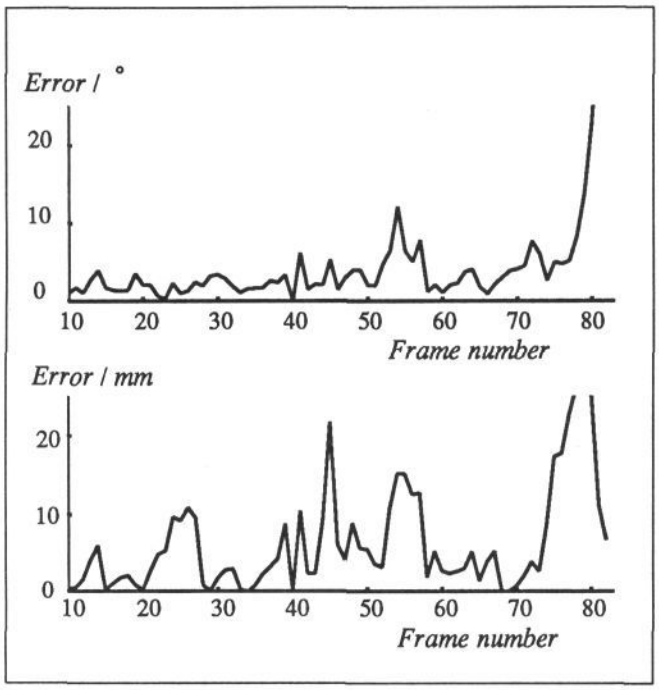

Figure 9. Plots of total output error in position and orientation against time.

object rotates clockwise, the estimated position is carried further away from the true position of the object. These results were obtained using 9x9 convolution, and three iterations of the Hough transform per frame. The RMS errors in position and orientation between frames 10 and 70 are 6.5mm and 3.4 degrees.

Whilst the accuracy of tracking is not all that it might have been, these results do demonstrate the robustness of the system in dealing with noise, occlusion and confusing backgrounds.

The implementation and performance of the system is described in more detail in [7].

CONCLUSIONS

I have described a system that performs unconstrained object tracking in real time. The computation is simple enough for it to be performed at frame rate using a small network of transputers. I attribute the speed, accuracy and robustness of the system to the technique of direct access to the image, making image segmentation unnecessary.

Much of the robustness of the system is due to the use of a Hough transform for solving constraints on the object's position. Hough transforms are also intrinsically parallel, and they provide an ideal way of combining information from two or more cameras. In dynamic Hough transforms, where accumulator cells are spherical, the use of a close-packed arrangement of cells has been proposed.

The transputer has been invaluable in the development of this project. It is the very high floating point performance of the transputer and its ability to form tightly coupled parallel systems that made it possible to consider a software implementation of the system.

ACKNOWLEDGEMENTS

My thanks are due to Prof. Frank Fallside (my PhD supervisor at Cambridge University), to Mark Wright and Eddie Murphy for their help in capturing the test sequence, and to Pablo Iglesias and Chris Blair.

REFERENCES

1. **Brisden, K.** "Alvey MMI-007 Vehicle Exemplar: Evaluation and Verification of Model Instances" *Proc. AVC 1987*

2. **Sharial, H. and Price, K. E.** "Results of Motion Estimation With More than Two Frames" *Proc. DARPA Image Understanding Workshop 1987*

3. **Liu, Y. and Huang, T. S.** "Estimation of Rigid Body Motion using Straight Line Correspondences" *IEEE Workshop on Motion: Representation and Analysis 1986*

4. **Canny, J. F.** "A Computational Approach to Edge Detection" *PAMI 8(6) pp 679-698 1986*

5. **Thompson, D. W. and Mundy, J. L.** "Three Dimensional Model Matching from an Unconstrained Viewpoint" *Proc. IEEE Conf. on Robotics and Automation 1987*

6. **Li, H, Lavin, M. A. and Le Master, R. J.** "The Fast Hough Transform: A Hierarchical Approach" *Computer Vision, Graphics and Image Processing 36 1986*

7. **Stephens, R. S.** "Implementation of a Real-Time 3D Object Tracker" *Proc. International conf. on Transputers for Industrial Applications II Oct 1989*

8. **Ahuja, D. V. and Coons, S. A.** "Geometry for Construction and Display" *IBM Systems Journal 7(3-4) pp 188-205 1968*

APPENDIX A: The Hough Transform

Homogenous Transforms [8] are used for mapping between coordinate systems.

The Mapping from world coordinates into image coordinates is given by the 3 by 4 camera matrix C, and the estimated position is M. $T = CM$ maps object coordinates into the image.

x is the 3D position of an edge point in object coordinates, and it is projected into the point u in the image, and n is a unit normal to the projection of the edge.

The transformation between estimated and true object position is given by the matrix P

$$P = \begin{bmatrix} 1 & -p_6 & p_5 & p_1 \\ p_6 & 1 & -p_4 & p_2 \\ -p_5 & p_4 & 1 & p_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where p_1 to p_6 are small parameters of the motion of the object.

$$p = [p_1, p_2, p_3, p_4, p_5, p_6]^T.$$

The edge displacement, d, is a function, f, of p.

$f(0) = 0$, since the edge displacement is theoretically zero if the estimated position coincides with the true position. Hence the Taylor's expansion of $f(p)$ is:

$$d \approx \left.\frac{df}{dp}\right|_{p=0} p = gp \tag{1}$$

Let $v(p)$ be the projection of the edge point from the true position, and let $y(p)$ be the 3D position of the edge point referred to the coordinate frame of the estimated position.

$$\begin{aligned} f(p) &= n \cdot (v(p) - u) \\ &= n \cdot v(p) - n \cdot u \\ \Longrightarrow \frac{df(p)}{dp} &= n \cdot \frac{dv(p)}{dp} \end{aligned} \tag{2}$$

Now,

$$Ty = \begin{bmatrix} sv_1 \\ sv_2 \\ s \end{bmatrix} \quad \text{where } v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

$$\Longrightarrow v_1 = \frac{t_1 y}{t_3 y}; \qquad v_2 = \frac{t_2 y}{t_3 y}$$

where t_1, t_2 and t_3 are the rows of T

$$\begin{aligned} \Longrightarrow \frac{dv_1}{dp} &= \frac{t_1 \frac{dy}{dp}}{t_3 y} - \frac{t_1 y}{(t_3 y)^2} t_3 \frac{dy}{dp} \\ &= \frac{1}{t_3 y}\left(t_1 - \frac{t_1 y}{t_3 y} t_3\right) F \end{aligned}$$

where

$$F = \frac{dy}{dp} = \begin{bmatrix} 1 & 0 & 0 & 0 & x_3 & -x_2 \\ 0 & 1 & 0 & -x_3 & 0 & x_1 \\ 0 & 0 & 1 & x_2 & -x_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\Longrightarrow \left.\frac{dv_1}{dp}\right|_{p=0} = \frac{1}{s}[t_1 - u_1 t_3]$$

similarly

$$\left.\frac{dv_2}{dp}\right|_{p=0} = \frac{1}{s}[t_2 - u_2 t_3] F$$

Hence, from (2), g may be found.

A Hough space cell has centre h and radius r. From (1), the cell is incremented iff

$$\begin{aligned} &|gh - d| \le r|g| \\ \Longrightarrow\ &gh - |g|r \le d \le gh + |g|r. \end{aligned}$$

This is the test that is performed for each edge displacement, for each Hough space cell. Note that gh and $|g|r$ can be computed before d has been measured, introducing parallelism between the workers and the framestore.

APPENDIX B: Close-packed Spheres.

A set of spheres close-packed in N-dimensional space can be generated as follows.

Definition:

A set of spheres close-packed about a central sphere satisfies the following conditions:

- all spheres touch the central sphere
- no two spheres intersect
- the set contains the maximum possible number of spheres.

The elements of a vector x are denoted $x[0], x[1], \ldots$

1. Generate the vectors P_0 to P_N as follows:

$$P_0 = 0$$

$$P_{i+1}[j] = \begin{cases} \frac{1}{i+1}\sum_{k=0}^{i} P_k[j] & \text{if } j < i \\ \sqrt{1 - \sum_{k=0}^{i} P_{i+1}^2[k]} & \text{if } j = i \\ 0 & \text{if } j > i \end{cases}$$

2. Then generate the sets of vectors S_0 to S_N as follows:

$$S_i = \{x : x = \pm(P_i - P_j), \forall j < i\}$$

3. The members of S_0 to S_N are the centre coordinates of a set of close–packed spheres of unit radius.

Example: $N = 4$

$$P0 = [0,0,0,0], \qquad P1 = [1,0,0,0]$$

$$P2 = [1/2, \sqrt{3}/2, 0, 0]$$

$$P3 = [1/2, 1/(2\sqrt{3}), \sqrt{2}/\sqrt{3}, 0]$$

$$P4 = [1/2, 1/(2\sqrt{3}), 1/(2\sqrt{2}\sqrt{3}), \sqrt{5}/(2\sqrt{2})]$$

$$S_0 = \{[0,0,0,0]\}$$

$$S_1 = \{\pm[1,0,0,0]\}$$

$$\begin{aligned} S_2 = \Big\{ &\pm\left[1/2, \sqrt{3}/2, 0, 0\right], \\ &\pm\left[-1/2, \sqrt{3}/2, 0, 0\right] \Big\} \end{aligned}$$

$$\begin{aligned} S_3 = \Big\{ &\pm\left[1/2, 1/(2\sqrt{3}), \sqrt{2}/\sqrt{3}, 0\right], \\ &\pm\left[-1/2, 1/(2\sqrt{3}), \sqrt{2}/\sqrt{3}, 0\right], \\ &\pm\left[0, -1/\sqrt{3}, \sqrt{2}/\sqrt{3}, 0\right] \Big\} \end{aligned}$$

$$\begin{aligned} S_4 = \Big\{ &\pm\left[1/2, 1/(2\sqrt{3}), 1/(2\sqrt{2}\sqrt{3}), \sqrt{5}/(2\sqrt{2})\right], \\ &\pm\left[-1/2, 1/(2\sqrt{3}), 1/(2\sqrt{2}\sqrt{3}), \sqrt{5}/(2\sqrt{2})\right], \\ &\pm\left[0, -1/\sqrt{3}, 1/(2\sqrt{2}\sqrt{3}), \sqrt{5}/(2\sqrt{2})\right], \\ &\pm\left[0, 0, -\sqrt{3}/(2\sqrt{2}), (\sqrt{5}/2\sqrt{2})\right] \Big\} \end{aligned}$$

NB, this is only one of many possible solutions.