Monograph:
Morris, A.S. and Deacon, G.E. (1986) Image -Based Visual Servo Control of a Revolute-Co-ordinate (Arm and Elbow) Robot. Research Report. Acse Report 288 . Dept of Automatic Control and System Engineering. University of Sheffield

---

IMAGE BASED VISUAL SERVO CONTROL OF A REVOLUTE-COORDINATE (ARM AND ELBOW) ROBOT

A. S. Morris, B.Eng., Ph.D., C.Eng, M.I.E.E., M.Inst. M.C.

G. E. Deacon, B.Sc., M.Eng.

Department of Control Engineering
University of Sheffield
Mappin Street
Sheffield S1 3JD

Research Report No. 288.

January 1986.

Abstract

A technique of using an image feature vector directly as the feedback control parameters in a robot driving strategy is described. This avoids the calculation of the inverse kinematic relationships involved in other robot driving algorithms based on robot vision systems, and so means a massive reduction in computational effort. The technique is applicable irrespective of the type of camera used in the vision system.

1. INTRODUCTION

Of all the industrial operations to which robots are broadly applicable, assembly tasks have by far the highest accuracy demands and present substantial problems to the robot system designer. Assembly operations require a component A held by a robot gripper to be correctly positioned and aligned with respect to a second component B. Thus the position and orientation of both the robot arm holding A, and of the component B, must be accurately known. Whilst it is possible to control the latter by the design and use of special jigs to constrain components, the former presents immense problems. Although it is possible to measure the position of robot joints by high resolution transducers such as optical encoders, there is no exact relationship between joint position and arm-end position because of joint backlash etc., effects. Conventional instrumentation is therefore unable to provide servo control of the robot gripper position, and alternative solutions are necessary. Robot vision systems provide the only known answer to this problem. When located such that a fixed geometrical relationship exists between the camera and gripper, a vision system can provide information about the relative position and orientation of the two components A and B. Besides overcoming the difficulty of achieving accurate servo control of robot arm position directly, this also obviates the need for special jigs to hold components.

Use of a robot vision system involves processing the image captured by the system and extracting information about the position and orientation of objects within the field of view. Common practice involves the computation of inverse knematic relationships in order to calculate the robot joint positions required to allow the component A held in the robot gripper to be assembled with component B. Unfortunately, such practice involves significant computational delays and also generates estimation noise.

An alternative technique is presented in this paper whereby image features are used directly as control parameters for a robot driving strategy involving a

zero-error seeking algorithm. The goal of such an algorithm is to drive the robot until the image of the target object, as viewed from the camera, is one which corresponds with the two components to be assembled being in the correct relative position and orientation. Essential to this technique is an accurate analysis of how the image feature vector changes as the distance and angle of approach between the gripper and target object change. A large saving in computational effort is made by such direct feedback of the image feature vector, and there is an additional advantage in terms of reduced estimation noise.

This technique represents a significant departure from previous practice, and merits the presentation of its theoretical aspects in advance of detailed discussion of the application considerations which will be described in a future paper after further work has been completed.

2. Image Analysis

The image captured by the camera in a robot vision system consists of an n x n pixel image array Λ, which is normally held in a framestore prior to processing. The image Λ is first programmed to another two-dimensional array Λp

i.e. $P: \Lambda \rightarrow \Lambda p$

where P describes the process of extracting the outline of objects within the field of vision of the image.

This is followed by a process F which extracts image features by mapping Λp into a one-dimensional feature vector $\underline{f}$

i.e. $F: \Lambda p \rightarrow \underline{f}$

The feature vector $\underline{f}$ consists of elements such as object area, perimeter, centroid, radius vectors etc. Interpretation of $\underline{f}$ yields position and orientation information $\underline{r}_c$ about the object B relative to the camera, which is conveniently described by a process Q.

i.e. $Q: \underline{f} \rightarrow \underline{r}_c$

The next conventional step S would be to use $\underline{r}_c$ to assess the attitude of the target object B in Euclidean space and compute the necessary joint positions to enable the robot end-effector holding component A to be moved such that A and B could be assembled together.

i.e. $S: \underline{r}_c \rightarrow \Theta$

where Θ is a vector of joint angles.

Unfortunately, process S involves inverse kinematic relationship calculations which require extensive and time-consuming computation.

This paper presents an alternative to steps Q and S by using the feature vector $\underline{f}$ directly as a feedback signal in the control process. This has an important advantage in terms of its much reduced computation requirement.

For this method, we assume that $\underline{f}$ is a smooth and bounded (though not necessarily monotonic function) of $\underline{r}_c$ and camera parameters s_c such that

$$\underline{f} = R(\underline{r}_c, \underline{s}_c) \qquad (1)$$

where R(.) may be analytically derived from a model of the object and an appropriate perspective transformation for the camera.
Features such as image areas, line lengths and corner angles satisfy this relationship and are often non-monotonic.

Having already mentioned the difficulty of measuring robot joint positions accurately, it is convenient to note that the position error information (the displacement between the gripper and the object) does not need to be very accurate as long as it is sufficient to provide convergence of the zero-error seeking control algorithm which actuates the control of the robot. If we have a reference feature vector, $\underline{f}_r$ say, and a feature vector derived from the image from purely geometrical considerations, $\underline{f}_i$ say, then a robot control signal can be generated from the difference between these two vectors. Some analysis of feature sensitivity is a necessary requirement in this process.

2.1 Feature Sensitivity Relationships

If we know how sensitive each feature is to a movement in the angle between the upper arm and the base of the robot, ϕ, and the angle between the upper and lower parts of the robot arm, ψ, then we can deduce how to move the robot links in order to reduce the difference between $\underline{f}_r$ and $\underline{f}_i$. The degree of sensitivity can be used to determine the length of time for which a particular link should be moved, since this is analogous to setting the gain for that link.

From purely geometrical considerations the values of the image feature vector will vary with ϕ, ψ, α and x_o, where α is the angle of orientation of the object (see fig.1) and x_o is the distance between the centre of the object and the shoulder of the object (see fig.2). So using the total differential theorem we have

$$df_j = \frac{\partial f_j}{\partial \phi}.d\phi + \frac{\partial f_j}{\partial \psi}.d\psi + \frac{\partial f_j}{\partial x_o}.dx_o + \frac{\partial f_j}{\partial \alpha}.d\alpha \qquad (2)$$

where $j = 1,2,\ldots,n$ and

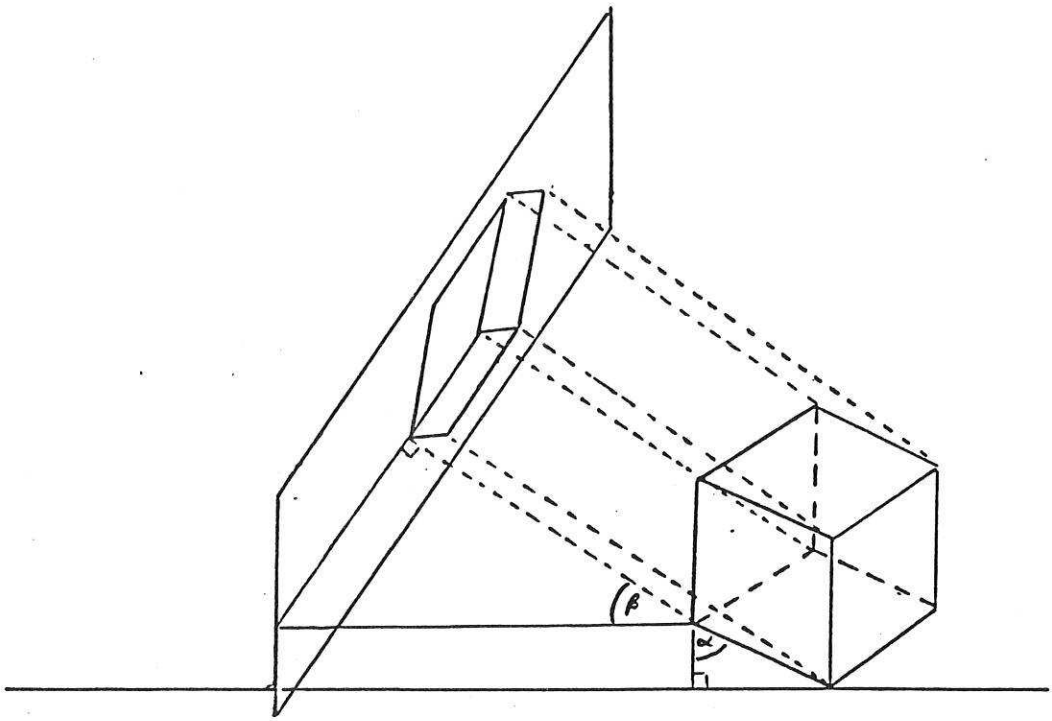

Figure 1. Diagram illustrating the angle of viewing β and the angle of orientation α with respect to the plane onto which the object is to be projected.

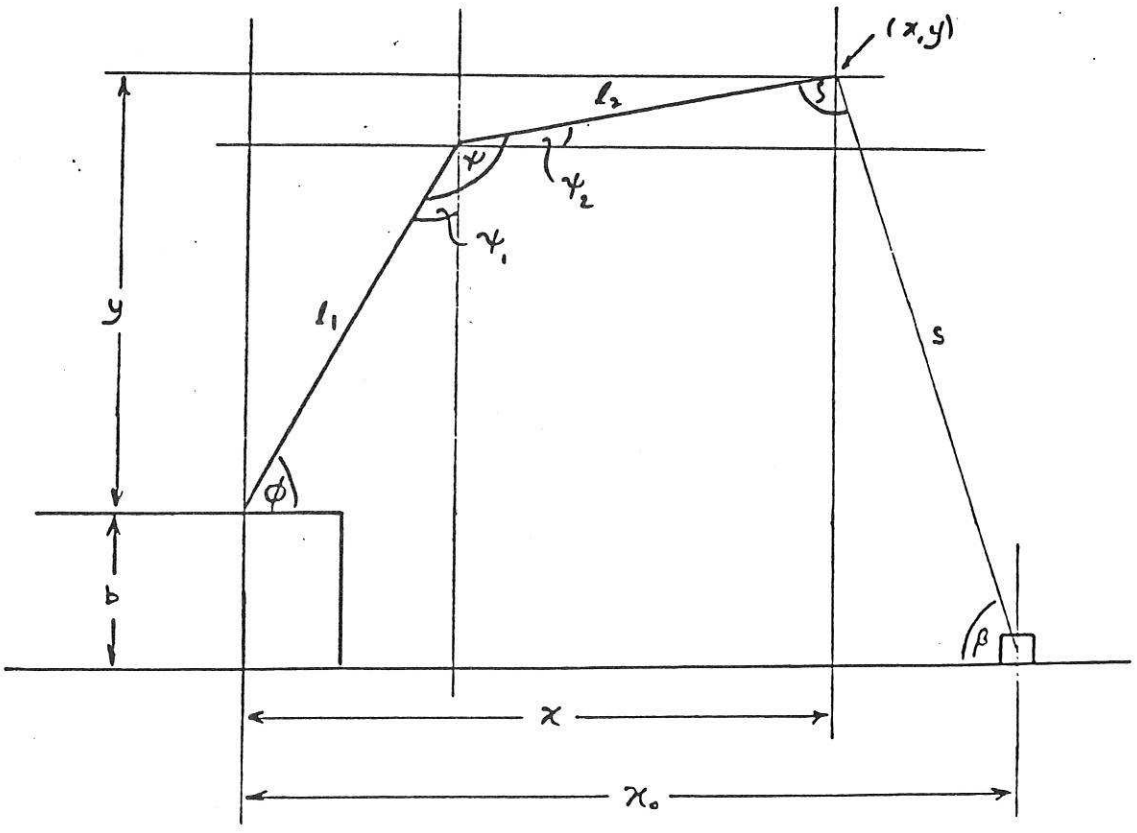

Figure 2. Geometrical model of the robot relative to an object (side view).

$$\underline{f}_i = \begin{pmatrix} f_1 \\ : \\ : \\ f_n \end{pmatrix}$$

But since x_o and α are constant for a given application, i.e.

$$dx_o = 0$$

$$d\alpha = 0$$

eqn (2) reduces to

$$df_j = \frac{\partial f_j}{\partial \phi} . d\phi + \frac{\partial f_j}{\partial \psi} . d\psi \qquad j = 1,2,...n \tag{3}$$

This can be written in matrix notation as

$$\underline{df} = D.G.\underline{dp}$$

where

$$\underline{df} = \begin{pmatrix} df_1 \\ : \\ : \\ df_n \end{pmatrix}$$ = vector of feature differentials

$$G = \begin{pmatrix} \frac{\partial f_1}{\partial \phi} & \frac{\partial f_1}{\partial \psi} \\ : & : \\ : & : \\ \frac{\partial f_n}{\partial \phi} & \frac{\partial f_n}{\partial \psi} \end{pmatrix}$$ = a Jacobian, known as the feature sensitivity matrix, which determines the sensitivity of the features for a given robot configuration.

$$\underline{dp} = \begin{pmatrix} d\phi \\ d\psi \end{pmatrix}$$ = vector of robot angle differentials

$D = \text{sign}(G)$ = the feature direction matrix and is employed since R(.) is not a monotonic function. It is used to determine the direction in which to drive the robot.

We now need an analytical solution for the differential coefficients of eqn (3). It is shown later in section 3 that

$$f_j = f_j(\beta, s)$$

for a given angle of orientation α, and fig. 2 illustrates that both β and s are function of ϕ and ψ i.e.

$$\beta = \beta(\psi, \phi)$$

$$s = s(\psi, \phi)$$

So, using the 'chain rule' we have

$$\left(\frac{\partial f_j}{\partial \phi}\right)_\psi = \left(\frac{\partial f_j}{\partial \beta}\right)_s \cdot \left(\frac{\partial \beta}{\partial \phi}\right)_\psi + \left(\frac{\partial f_j}{\partial s}\right)_\beta \cdot \left(\frac{\partial s}{\partial \phi}\right)_\psi \tag{4}$$

and

$$\left(\frac{\partial f_j}{\partial \psi}\right)_\phi = \left(\frac{\partial f_j}{\partial \beta}\right)_s \cdot \left(\frac{\partial \beta}{\partial \psi}\right)_\phi + \left(\frac{\partial f_j}{\partial s}\right)_\beta \cdot \left(\frac{\partial s}{\partial \psi}\right)_\phi \tag{5}$$

But β and s are implicit functions of ψ and ϕ since

$$\beta = \beta'(x,y)$$

$$s = s'(x,y)$$

$$x = (\psi, \phi)$$

$$y = (\psi, \phi)$$

where (x,y) is the position of the wrist end, of the lower arm of the robot, relative to the shoulder joint. Hence by applying the chain rule again we arrive at the following equations

$$\left(\frac{\partial \beta}{\partial \phi}\right)_\psi = \left(\frac{\partial \beta}{\partial x}\right)_y \cdot \left(\frac{\partial x}{\partial \phi}\right)_\psi + \left(\frac{\partial \beta}{\partial y}\right)_x \cdot \left(\frac{\partial y}{\partial \phi}\right)_\psi \tag{6}$$

$$\left(\frac{\partial \beta}{\partial \psi}\right)_\phi = \left(\frac{\partial \beta}{\partial x}\right)_y \cdot \left(\frac{\partial x}{\partial \psi}\right)_\phi + \left(\frac{\partial \beta}{\partial y}\right)_x \cdot \left(\frac{\partial y}{\partial \psi}\right)_\phi \tag{7}$$

and

$$\left(\frac{\partial s}{\partial \phi}\right)_\psi = \left(\frac{\partial s}{\partial x}\right)_y \cdot \left(\frac{\partial x}{\partial \phi}\right)_\psi + \left(\frac{\partial s}{\partial y}\right)_x \cdot \left(\frac{\partial y}{\partial \phi}\right)_\psi \tag{8}$$

$$\left(\frac{\partial s}{\partial \psi}\right)_\phi = \left(\frac{\partial s}{\partial x}\right)_y \cdot \left(\frac{\partial x}{\partial \psi}\right)_\phi + \left(\frac{\partial s}{\partial y}\right)_x \cdot \left(\frac{\partial y}{\partial \psi}\right)_\phi \tag{9}$$

2.2 Solution of the Feature Sensitivity Equations

The solution of equations (4) - (9) is facilitated by reference to Fig. 2 from which we have

$$\tan\beta = \frac{y + b}{(x_o - x)^2}$$

Hence

$$\sec^2\beta . \left(\frac{\partial\beta}{\partial x}\right)_y = \frac{y + b}{(x_o - x)^2}$$

$$\left(\frac{\partial\beta}{\partial x}\right)_y = \frac{y + b}{(x_o - x)^2} . \cos^2\beta$$

But since we do not want to measure x we can use the fact that

$$\cos\beta = \frac{(x_o - x)}{s}$$

hence

$$\left(\frac{\partial\beta}{\partial x}\right)_y = \frac{y + b}{s^2} = \frac{\sin\beta}{s} \qquad (10)$$

since

$$\sin\beta = \frac{y + b}{s}$$

Similarly

$$\sec^2\beta . \left(\frac{\partial\beta}{\partial y}\right)_x = \frac{1}{x_o - x}$$

$$\left(\frac{\partial\beta}{\partial y}\right)_x = \frac{\cos^2\beta}{x_o - x} = \frac{\cos\beta}{s} \qquad (11)$$

From fig.2 again, we have the following relationships

$$x = l_1.\cos\phi + l_2.\cos\psi_2$$

$$y = l_1.\sin\phi + l_2.\sin\psi_2$$

But since

$$\psi = \psi_2 + 90^o + \psi_1$$

and

$$\psi_1 = 90^o - \phi$$

we have

$$\psi_2 = \psi + \phi - 180^o$$

from which it follows that

$$x = l_1.\cos\phi + l_2.\cos(\psi + \phi - 180^o)$$

$$= l_1.\cos\phi - l_2.\cos(\psi + \phi) \qquad (12)$$

$$y = l_1.\sin\phi + l_2.\sin(\psi + \phi - 180^o)$$

$$= l_1.\sin\phi - l_2. \sin(\psi + \phi) \qquad (13)$$

From equations (12) and (13) we can deduce the following results

$$\left(\frac{\partial x}{\partial \phi}\right)_{\psi} = -l_1,\sin\phi + l_2.\sin(\psi + \phi) = -y \tag{14}$$

$$\left(\frac{\partial x}{\partial \psi}\right)_{\phi} = l_2.\sin(\psi + \phi) \tag{15}$$

$$\left(\frac{\partial y}{\partial \phi}\right)_{\psi} = l_1.\cos\phi - l_2.\cos(\psi + \phi) = x \tag{16}$$

$$\left(\frac{\partial y}{\partial \psi}\right)_{\phi} = -l_2.\cos(\psi + \phi) = x \tag{17}$$

Also, from fig.2 we have (using Pythagoras)

$$s^2 = (y + b)^2 + (x_o - x)^2$$

Hence we obtain

$$2s.\left(\frac{\partial s}{\partial x}\right)_y = -2(x_o - x)$$

$$\left(\frac{\partial s}{\partial x}\right)_y = -\frac{(x_o - x)}{s} = -\cos\beta \tag{18}$$

and

$$2s.\left(\frac{\partial s}{\partial y}\right)_x = 2(y + b)$$

$$\left(\frac{\partial s}{\partial y}\right)_x = \frac{y + b}{s} = \sin\beta \tag{19}$$

There are still two partial derivatives not yet calculated - those of $\left(\frac{\partial f_j}{\partial \beta}\right)_s$ and $\left(\frac{\partial f_j}{\partial s}\right)_\beta$. To evaluate these we first need to find expressions for the f_j in terms of β and s. These expressions will be derived in the following section.

To summarise, we have so far derived the following results:

$$\left(\frac{\partial \beta}{\partial \phi}\right)_{\psi} = \left(\frac{\sin\beta}{s}\right) . (-y) + \left(\frac{\cos\beta}{s}\right) .(x) \tag{20}$$

$$\left(\frac{\partial \beta}{\partial \psi}\right)_{\phi} = \left(\frac{\sin\beta}{s}\right) . [l_2. \sin(\psi + \phi)] + \left(\frac{\cos\beta}{s}\right) . [-l_2.\cos(\psi + \phi)] \tag{21}$$

$$\left(\frac{\partial s}{\partial \phi}\right)_{\psi} = -(\cos\beta).(-y) + (\sin\beta).(x) \tag{22}$$

$$\left(\frac{\partial s}{\partial \psi}\right)_{\phi} = (-\cos\beta).[l_2.\sin(\psi + \phi)] + (\sin\beta).[l_2.\cos(\psi + \phi)] \tag{23}$$

2.3 Analysis of Orthographic projections

In this section, the derivation of the function $R(\underline{r}_c, \underline{s}_c)$ of eqn. (1) is investigated. In order to simplify the development of the concepts associated with image analysis, the following analysis is restricted to the problem of picking up a cube, this being a geometrically simple shape. Having thus developed a suitable method for the particular case of a cube, the necessary theoretical framework has been built to allow the image analysis scheme to be extended to more complex polyhedra representing the general case.

The situation under consideration is shown in fig.1. where α is the angle of orientation and β is the angle of viewing. The variation of the orthographic projection with α and β is shown in fig.3. It can readily be seen that the features derived by image analysis, such as perimeter length, area, minimum enclosing rectangle etc. vary with both the angle of viewing, and the angle of orientation. The question is, exactly how do these features vary with α and β ?

If we consider a cube from an arbitrary angle of viewing and an arbitrary orientation, we would have an orthographic projection which typically looks similar to fig.4. Some of the relationships between this projection and the original object will now be derived.

(a) Minimum Enclosing Rectangle

Referring to fig.5, for a cube of side length L , we have

$$\begin{aligned} N &= L_1 + L_2 \\ &= L_o.\sin\alpha + L_o.\cos\alpha \\ &= L_o(\sin\alpha + \cos\alpha) \end{aligned} \tag{24}$$

Referring to fig.6, the length of μ is given by

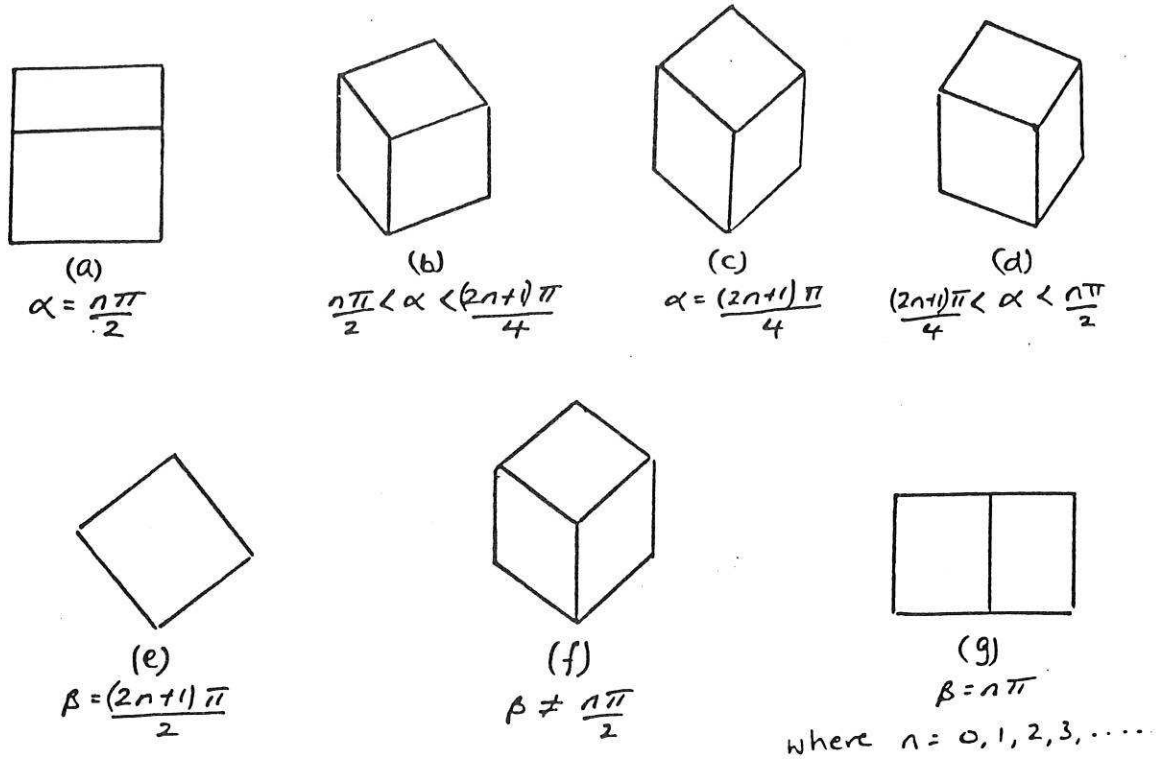

Figure 3. The variation of the orthographic projection with α for a given β, and with β for a given α.

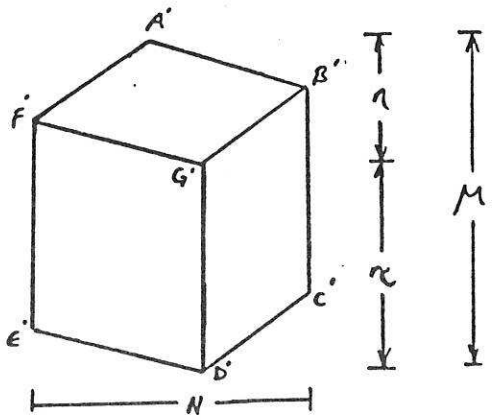

Figure 4. A typical orthographic projection of a cube.

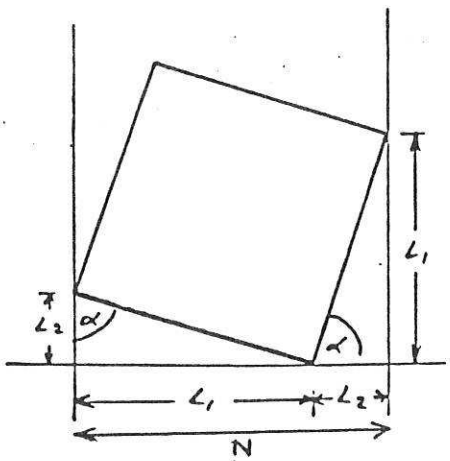

Figure 5. Plan view of a cube at orientation α to the projection plane.

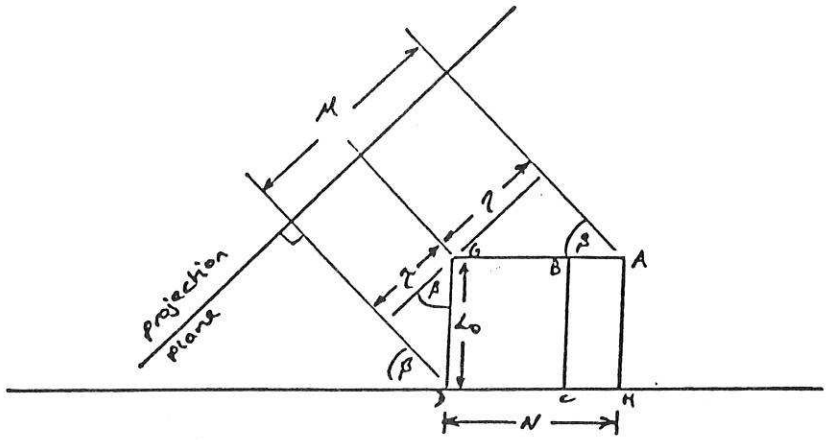

Figure 6. Side view of a cube at an angle β to the projection plane.

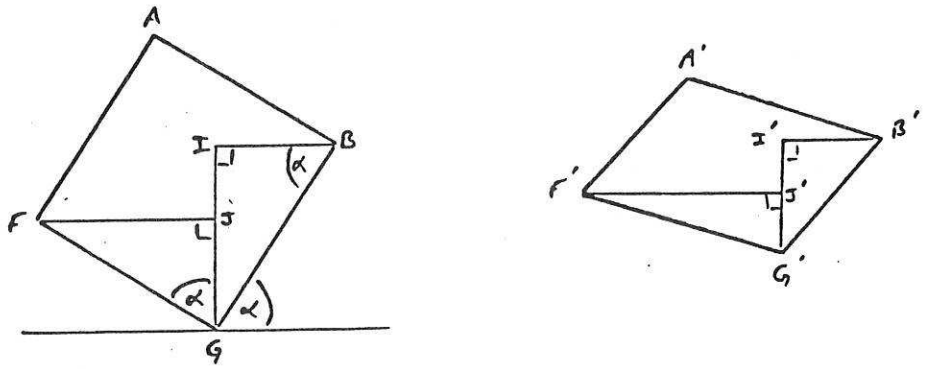

Figure 7. The plan view of a cube and its projected image.

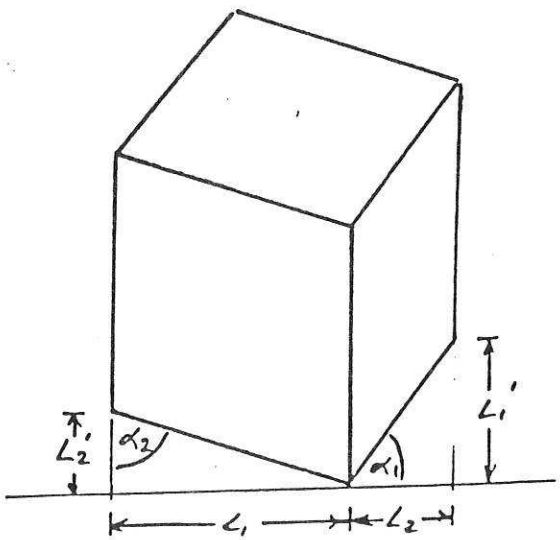

Figure 8. The definition of angles α_1 and α_2.

$$\mu = \eta + \tau$$

$$= N.\sin\beta + L_o.\cos\beta$$

$$= L_o\left[(\sin\alpha + \cos\alpha)\sin\beta + \cos\beta\right] \qquad (26)$$

N and μ represent the width and height respectively of the minimum enclosing rectangle around the image of an object twisted through an angle α in the vertical plane and viewed at an angle β with respect to the horizontal plane.

(b) Perimeter

The perimeter of the orthographic projection is given by the sum of the projected lengths of sides AB, BC, CD, DE, EF and FA.

From fig.6.

$$G'D' = \tau = GD.\cos\beta + L_o.\cos\beta \qquad (27)$$

To derive expressions for the projected lengths of AB, CD, DE and FA consider the plan view of the cube shown in fig.7.

The triangle HBG is transformed to the triangle H'B'G', where

$$I'B' = IB$$

$$I'G' = IG.\sin\beta$$

Hence the line B'G', the projected length of BG, is given by Pythagoras as

$$(B'G')^2 = (I'G')^2 + (I'B')^2$$

Substituting for I'G' we get

$$(B'G')^2 = (IG.\sin\beta)^2 + (IB)^2$$

But from fig.7

$$IG = BG.\sin\alpha$$

$$IB = BG.\cos\alpha$$

and since

$$BG = L_o$$

we have

$$(B'G')^2 = L_o^2.\sin^2\alpha.\sin^2\beta + L_o^2.\cos^2\alpha$$

$$B'G' = L_o\sqrt{\sin^2\alpha.\sin^2\beta + \cos^2\alpha}$$

Using the identity

$$\sin^2\theta + \cos^2\theta = 1$$

this reduces to

$$B'G' = L_o \sqrt{1 - \sin^2\alpha.\cos^2\beta} \quad (28)$$

From similar considerations we also have

$$F'G' = L_o \sqrt{1 - \cos^2\alpha.\cos^2\beta} \quad (29)$$

The length of the perimeter now follows from the fact that

$$A'B' = F'G' = E'D'$$

$$A'F' = B'G' = C'D'$$

$$F'E' = G'D' = B'C'$$

Hence, the perimeter length

$$= 2L_o \left(\cos\beta + \sqrt{1 - \sin^2\alpha.\cos^2\beta} + \sqrt{1 - \cos^2\alpha.\cos^2\beta}\right) \quad (30)$$

(c) Angles

The angles of an object are a very useful addition to the feature set since their dependence on α and β can be readily seen from fig.3.

With reference to fig.5, it can be seen that the angle α is defined by the line lengths L_1 and L_2. Similarly, the angles α_1 and α_2 (fig.8) are defined by the line lengths L_1' and L_2, and L_1 and L_2' respectively. It follows from the results for the minimum enclosing rectangle that

$$L_1' = L_1.\sin\beta \quad (31)$$

$$L_2' = L_2.\sin\beta \quad (32)$$

From fig.8 we have

$$\tan\alpha_1 = \frac{L_1'}{L_2} = \frac{L_1.\sin\beta}{L_2}$$

$$\tan\alpha_2 = \frac{L_1}{L_2'} = \frac{L_1}{L_2.\sin\beta}$$

and from fig.5.

$$\tan\alpha = \frac{L_1}{L_2}$$

Combining these three equations gives

$$\tan\alpha_1 = \tan\alpha.\sin\beta \quad (33)$$

$$\tan\alpha_2 = \tan\alpha.\operatorname{cosec}\beta \quad (34)$$

i.e. angles α_1 and α_2 can be expressed as functions of α and β only, and since from geometric considerations the angles of the orthographic projection can be expressed in terms of α_1 and α_2 only (see fig.9), then the angles seen in the image are directly related to the orientation of the object and the angle of viewing.

(d) Area

The total projected area can be calculated from the summation of the projected areas of the three visible faces. So, with reference to fig.9.

$$\text{Area of face 1} = (L_1' + L_2').(L_1 + L_2) - [(L_1'.L_2) + (L_1.L_2')]$$
$$= L_1'.L_1 + L_2'.L_2$$

$$\text{Area of face 2} = L_1.G'D'$$

$$\text{Area of face 3} = L_2.G'D'$$

Substituting for L_1', L_2' and $G'D'$ from equations (31), (32) and (27) respectively, and using the fact that from fig.8. we can deduce

$$L_1 = L_o.\sin\alpha$$

$$L_2 = L_o.\cos\alpha$$

we arrive at

$$\text{Area of face 1} = L_1^2.\sin\beta + L_2^2.\sin\beta$$
$$= L_o^2.\sin\beta(\sin^2\alpha + \cos^2\alpha)$$
$$= L_o^2.\sin\beta \qquad (35)$$

$$\text{Area of face 2} = L_o^2.\sin\alpha.\cos\beta \qquad (36)$$

$$\text{Area of face 3} = L_o^2.\cos\alpha.\cos\beta \qquad (37)$$

So the total area of the projection is

$$= L_o^2[\sin\beta + (\sin\alpha + \cos\alpha)\cos\beta] \qquad (38)$$

Notice that the area of face 1 depends only on L_o, which is known a priori, and β. Consequently, if the area of face 1 could be found by using a method of lighting the object sequentially from more than one direction, we would have an estimate of β.

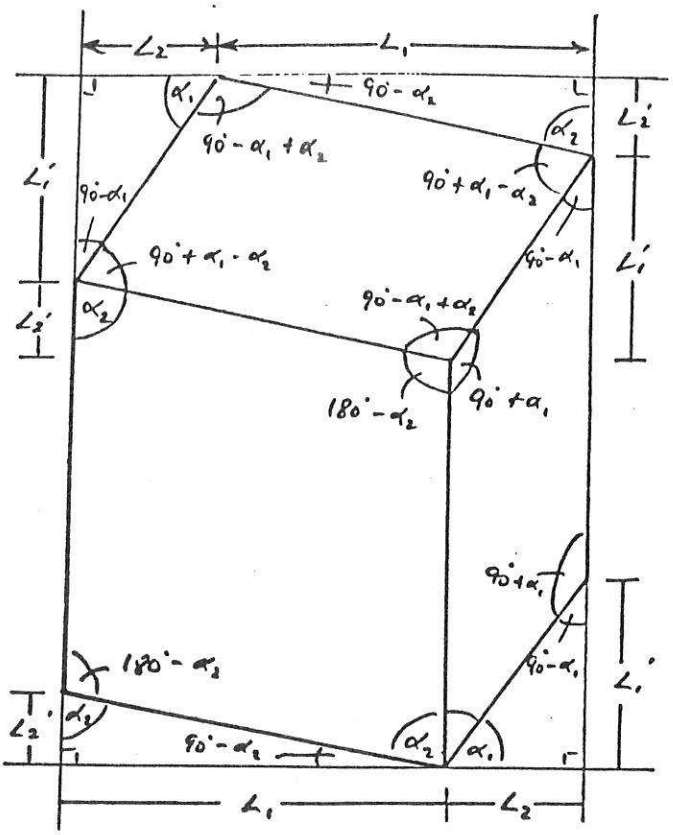

Figure 9. Diagram showing how all the angles of the orthographic projection can be expressed in terms of α_1 and α_2.

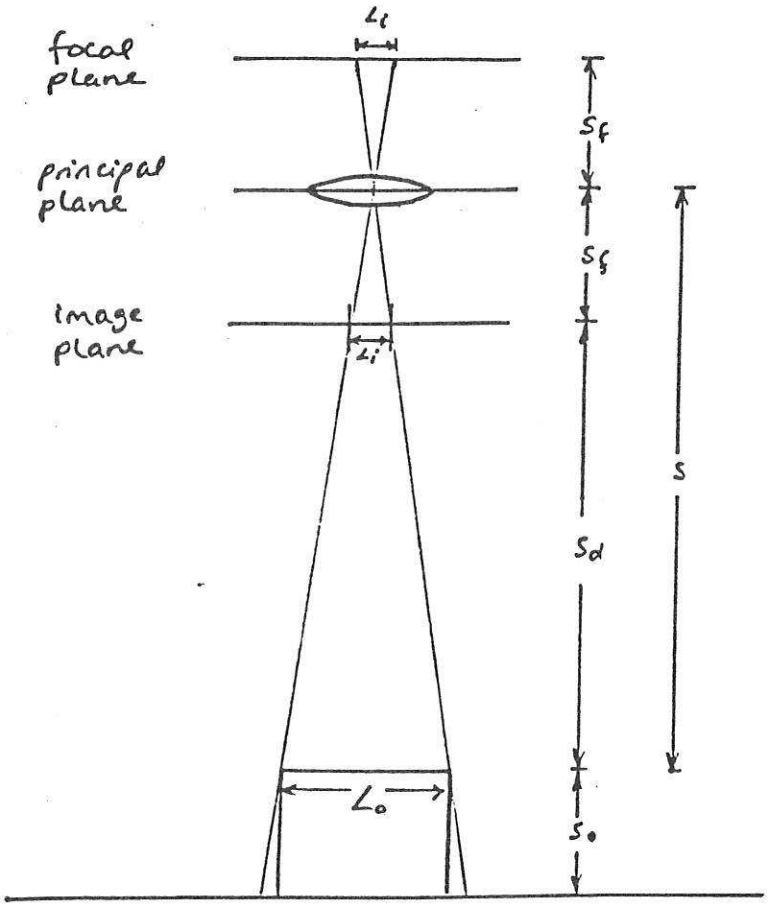

Figure 10. Perspective projection of the object due to the camera

(e) Centre of Area and Radius Vectors

The centre of area and radius vectors are further aspects of the feature set which could now be calculated. Such calculations are substantially more complicated however than for those elements of the feature set discussed so far, and they are not of immediate importance to the theoretical framework being developed for the particular case of the geometrically simple cube. These features are not therefore considered here, but they will of course have relevance in extensions of the theoretical framework being developed to more geometrically complicated polyhedra.

2.4 Perspective Projections

If the orthographic projection described above was the only transformation of the image involved, then we would always see the object as the same size no matter what distance it was viewed from. However, it is apparent that the further the object gets from the lens (of any vision system), the smaller the object becomes. The reason for this is apparent from fig.10 since the greater s gets, the smaller L_i becomes.

The perspective transformation of the camera, necessary for the derivation of the function $R(\underline{r}_c, \underline{s}_c)$ of equation (1), can be derived from fig.10, where by similar triangles we have

$$\frac{L_i}{s_f} = \frac{L_o}{s} \tag{39}$$

and upon rearranging

$$s = \frac{L_o . s_f}{L_i} \tag{40}$$

In equation (40), L_o is known a priori, L_i can be measured from the image and s_f is the known focal length of the camera. Hence we can get an estimate of s, the distance of the object from the camera, from the size of the image.

3. Implementation considerations

The last section has examined how features of the object are transformed to features of the image. What remains now is to relate feature sensitivity

to changes in the angles of ϕ and ψ (see fig.2) and consider how to implement this in a feedback algorithm.

3.1 The Sensitivity of the Height of the Minimum Enclosing Rectangle to changes in Robot Position

From the results of the previous section we have

$$\mu_o = L_o[(\cos\alpha + \sin\alpha)\sin\beta + \cos\beta] \tag{24}$$

where μ_o is the height of the minimum enclosing rectangle of the orthographic projection of the object. We also have

$$\frac{\mu_i}{\mu_o} = \frac{s_f}{s} \tag{41}$$

which follows from equation (39) where μ_i is the height of the minimum enclosing rectangle in the image. Combining equations (24) and (41) we arrive at

$$\mu_i = \frac{s_f.L_o}{s}.[(\cos\alpha + \sin\alpha)\sin\beta + \cos\beta] \tag{42}$$

In order to calculate $\left(\frac{\partial\mu_i}{\partial\phi}\right)_\psi$ and $\left(\frac{\partial\mu_i}{\partial\psi}\right)_\phi$ we first need to find $\left(\frac{\partial\mu_i}{\partial\beta}\right)_s$ and $\left(\frac{\partial\mu_i}{\partial s}\right)_\beta$ in addition to equations (20) and (23); these can be calculated from:-

$$\left(\frac{\partial\mu_i}{\partial\beta}\right)_s = \frac{s_f.L_o}{s}.[(\cos\alpha + \sin\alpha)\cos\beta - \sin\beta] \tag{43}$$

$$\left(\frac{\partial\mu_i}{\partial s}\right)_\beta = \frac{-s_f.L_o}{s^2}.[(\cos\alpha + \sin\alpha)\sin\beta - \cos\beta] \tag{44}$$

Now upon substituting equations (16) – (19) and equations (43) and (44) into equations (1) and (2) we arrive at

$$\left(\frac{\partial\mu_i}{\partial\phi}\right)_\psi = \frac{s_f.L_o}{s}.[(\cos\alpha + \sin\alpha)\cos\beta - \sin\beta].\frac{-y.\sin\beta}{s} + \frac{x.\cos\beta}{s} + \frac{-\mu_i}{s}.(y.\cos\beta + x.\sin\beta) \tag{45}$$

and

$$\left(\frac{\partial\mu_i}{\partial\psi}\right)_\phi = \frac{s_f.L_o}{s}.[(\cos\alpha + \sin\alpha)\cos\beta - \sin\beta].$$
$$[-l_2.\cos\beta.\sin(\psi + \phi) - l_2.\sin\beta.\cos(\psi + \phi)]$$

$$+\frac{-\mu_i}{s}\cdot\left[-l_2.\cos\beta.\sin(\psi+\phi)\ -\ l_2.\sin\beta.\cos(\psi+\phi)\right] \qquad (46)$$

We should know s_f, L_1, l_1 and l_2 a priori, μ_i from the image, and ought to be able to get an estimate of α from equation (24), an estimate of β from equation (35) and an estimate of s from equation (40). Since x and y can both be expressed in terms of ϕ and ψ (equations (12) and (13)) the only unknowns in equations (45) and (46) are ϕ and ψ.

3.2 Robot Driving Algorithm

One possible robot driving strategy is to pick arbitrary values for ϕ and ψ in the middle of the range of motion fo each joint, which we shall call ϕ_a and ψ_a. We can then calculate the sensitivity of features for the robot configuration at angles ϕ_a and ψ_a. The product of the error generated between a reference and observed feature, and the inverse of the feature sensitivity to a change in either ϕ or ψ, will be proportional to the length of time each link should move for. The sign of the error will determine the direction.

Obviously this is not an optimum solution since the features' sensitivity will vary depending on ϕ and ψ. The system would have to work on the first image received, move, and then re-assess how to move subsequently. Owing to the number of estimates involved, and the fact that the image involved will not be perfect, it may possibly be a long process before the error converges to zero.

A better solution is to calibrate joint position transducers and use them to get estimates of ϕ and ψ. The various feature sensitivities for different configurations can then be stored in a look-up table and can be used to get a more appropriate estimate of how the robot links should be driven, rather than basing the whole control algorithm on the feature sensitivities at an arbitrary point.

3.3 Implementation

It has been assumed in the above discussion that the wrist joint would be used to relocate the object as the upper and lower arm links move. It has also been implicitly assumed in the preceeding analysis that the camera would be mounted at the point (x,y) of fig.2. This would not normally be possible in practice

and the camera would have to be mounted close to (x,y) rather than exactly on it. It is probably worth mentioning that being mounted so close to the axis of rotation of the wrist would make the change in scene viewed by the camera very sensitive to a change in the angle between the wrist and the lower arm (ζ in fig.2.).

In the analysis of section 2.3, the projection of the object is nearly always shown with six corners around the perimeter. However it is possible that the image may include a projection like that shown in fig. 3(a). Indeed the reference feature vector would probably describe the image of fig. 3(e) i.e. we want the gripper looking directly down onto the object. Hence the first feature we ought to look for is the number of corners seen. If six corners are initially seen the control algorithm ought to work on, for example, reducing the lengths of lines E'F' and B'C' of Fig.4 until four corners are seen.

If the image has four corners, the control algorithm can work on the ratio of the side lengths until all the side lengths are the same length. (For polyhedra other than a cube the side lengths may not be a good choice of feature for determining whether the camera is directly above the object or not - see Appendix 1). Even if four corners are observed, and all the side lengths are the same, the area taken up by the object in the camera image would have to be of a predetermined size before the gripper can be considered in a position suitable to pick the object up. It should be possible to drive the robot with all these considerations taken into account simultaneously in order to arrive at some kind of optimum trajectory.

Since the camera will have to be mounted next to the wrist, it is the camera that will be directly above the object, not the gripper, when the desired image is seen. It will be necessary to move the robot arm by a known offset in order to get the gripper in the appropriate position. To do this we would need to know the orientation of the wrist at the point of pick up.

If the joint positions are to be estimated as suggested, it may be advisable to stop the robot to perform the estimation and assess an image at this point. This would eliminate the problem of the blurred image and the control algorithm would have a joint estimate along with the corresponding image. Additionally, due to the amount of time it takes to process an image at the moment, it is unlikely that the robot will be pressed to 'keep up' with the image processing.

A block diagram illustrating the suggested system is shown in fig.11.

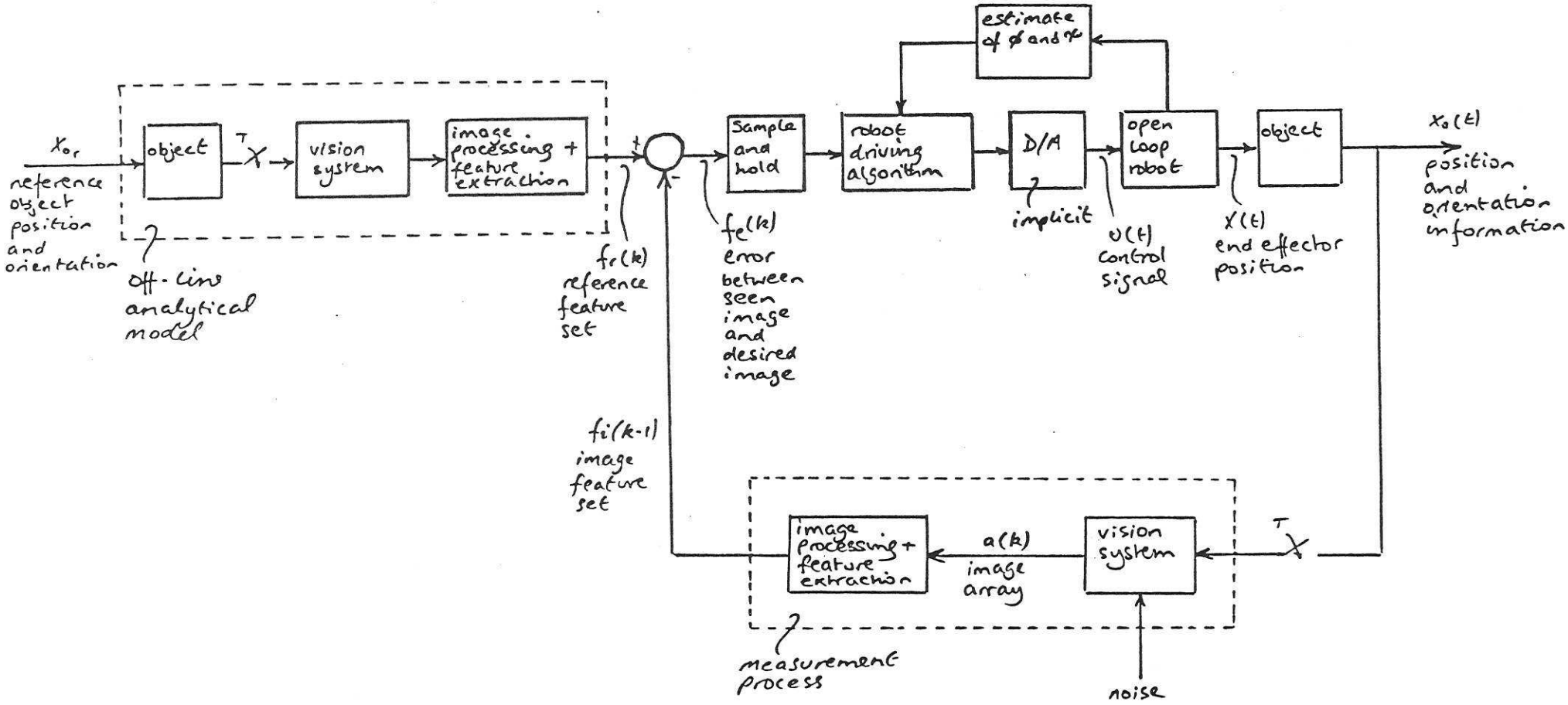

Figure 11. A block diagram illustrating the suggested image-based visual feedback scheme.

APPENDIX 1.

Appendix 1 - Ambiguity arising from using side length as an indication of viewing angle for certain polyhedra.

For an object such as that shown in fig.A1, which is square on to the projection plane (i.e. four corners show in the orthographic projection), there is a certain value of β where the image appears the same as would be seen when viewing from directly above. This occurs when the length μ in the orthographic projection equals the length N of the object, since the width of the object in the orthographic projection will be unchanged. The following analysis calculates the value of β at which this occurs in relation to the dimensions of the object.

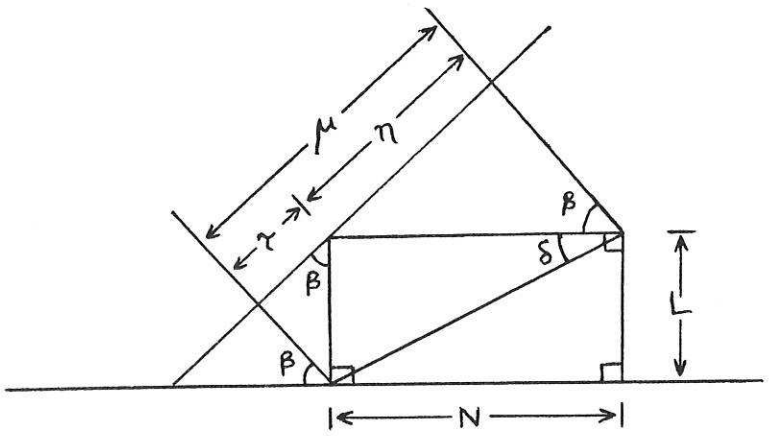

Figure A1. The object under consideration in relation to the projection plane.

From fig.A1 we have

$$\mu = \eta + \tau$$

$$= N.\cos\beta + L.\sin\beta$$

Dividing through by $\sqrt{L^2 + N^2}$ gives

$$\frac{\mu}{\sqrt{L^2 + N^2}} = \frac{N.\cos\beta}{\sqrt{L^2 + N^2}} + \frac{L.\sin\beta}{\sqrt{L^2 + N^2}} \qquad (47)$$

but

$$\cos\delta = \frac{N}{\sqrt{L^2 + N^2}} \qquad (48)$$

and

$$\sin\delta = \frac{L}{\sqrt{L^2 + N^2}} \tag{49}$$

So equation (47) becomes

$$\frac{\mu}{\sqrt{L^2 + N^2}} = \cos\delta.\cos\beta + \sin\delta.\sin\beta$$
$$= \cos(\beta - \delta)$$

Now we are interested in when $\mu = N$, i.e. when

$$\frac{N}{\sqrt{L^2 + N^2}} = \cos(\beta = \delta)$$

or

$$\text{arc.cos} \frac{N}{\sqrt{L^2 + N^2}} + \delta = \beta$$

which upon substitution for δ from equation (48) gives the value for β at which the object would appear to be viewed from directly above as

$$\beta = 2\delta$$

Note that for a cube, δ is 45^{o}, so the image appears the same as when the object is viewed from directly above (when $\beta = 90^{o}$).