Vision Guided Circumnavigating Autonomous Robots *

Nick Barnes and Zhi-Qiang Liu
Computer Vision and Machine Intelligence Lab,
Department of Computer Science
The University of Melbourne,
Parkville, Victoria, 3052, AUSTRALIA

Abstract

We present a system for vision guided autonomous circumnavigation, allowing a mobile robot to navigate safely around objects of arbitrary pose, and avoid obstacles. The system performs model-based object recognition from an intensity image. By enabling robots to recognise and navigate with respect to particular objects, this system empowers robots to perform deterministic actions on specific objects, rather than general exploration and navigation as emphasised in much of the current literature. This paper describes a fully integrated system, and, in particular, introduces canonical-views. Further, we derive a direct algebraic method for finding object pose and position for the four-dimensional case of a ground-based robot with uncalibrated vertical movement of its camera. Vision for mobile robots can be treated as a very different problem to traditional computer vision, as mobile robots have a characteristic perspective, and there is a causal relation between robot actions and view changes. Canonical-views are a novel, active object representation designed specifically to take advantage of the constraints of the robot navigation problem to allow efficient recognition and navigation.

*This work is supported in part by an Australian Research Council (ARC) large grant.

1. Introduction

Mobile robot guidance systems generally focus on problems of navigating through an environment, often by localising with respect to the environment, and moving according to a global map, (e.g., Refs. [11,30,36]). Objects are treated as obstacles to be avoided. Few systems in the literature examine the problem of recognising and interacting with objects. However, many tasks that mobile robots may perform (e.g., in dangerous environments, manufacturing, mining, and agricultural applications) involve identifying a specific object and performing a predetermined interaction with it. In order to support interactions such as looking for faults or markings on a particular surface, or manipulating part of the object (e.g., a handle or control panel), a robot must first identify the object, and then move with respect to it to the point where the interaction is required. In this paper, we present a system which uses model-based object recognition to uniquely identify a specified object, and moves around the object maintaining knowledge of its relative position. We call this task circumnavigation, and present it as a framework to support interaction. Existing research on docking[34,35], and manipulation[25,31], supports interaction, but only when the robot is in a position close to the required point of interaction.

In object recognition, the dominant research approach is to recognise from single, static images, taken from any random view, with no defined task. We introduce an approach that is specific for mobile robots and emphasise the *causal* nature of interaction, particularly, changes in a robot's perception of a stationary object are caused by robot actions. Our system focuses its attention on the object with which it is required to interact. Other visible features are regarded as background, and non-target objects are possible obstacles. Since, robot navigation systems are task-directed by definition, we use constraints of the robot navigation task to direct our system's perception, and make the object model more efficient. We introduce a novel representation called canonical-views: an active, viewer-centred object model, specifically designed for robot navigation. Canonical-views represent what the robot can see in practice, rather than what theoretically appears. Further, once an object has been recognised, the canonical-view model predicts which view will appear next, so the robot generally only has to match a single view of the object of interest, allowing fast recognition. The system does not require an environment model, as it navigates with respect to the required object, and can detect and avoid obstacles as it moves.

Once an object has been identified, this circumnavigation system can move around the object to any point relative to the object. While, tracking (e.g., Ref. [22]) and visual servoing (e.g., Ref. [21]) techniques can track objects features, current methods seldom offer an approach for: initially finding features; recovery if features shift suddenly between frames; or, recovery if features are lost briefly due to specularities or occlusion. When such systems do initialise features it is typically using assumptions that do not apply generally, such as velocity segmentation[21], which relies on the target being the only object moving in the field of view. Some active vision systems consider non-affine object transformations[23]. However, as a robot moves around an object, surfaces rotate relative to the camera and become occluded and new surfaces emerge. Most tracking systems are ill-equipped for this situation. By using edge matching, and object model

knowledge of which surface will appear next, our system is able to maintain its gaze on an object as it moves around it, and the surfaces it was following become completely occluded, and new surfaces become visible.

1.1. Related Research

Dickinson *et al.*[14] model objects for recognition as compounds of a finite set of volumetric part classes, simplifying object model indexing. In Ref. [15] this method is applied to recognise objects for the purpose of guiding a robot arm. However, this method is only applicable to objects composed of distinct volumetric parts[12,13].

Kutulakos and Dyer[29] purposefully control the motion of an active monocular observer to recover a global description of an object's shape. For our system, the purposive approach suggests the robot should take the shortest path to its required location. This has been demonstrated in earlier work.[4] In low-level vision tasks, (e.g., visual servoing), the computational cost of transformation into 3D space is often not justified as many tasks can be handled in image space. However, operating in 3D space allows our system to use 3D spatial knowledge for vision processing and navigation, enabling this type of purposive action.

Both of the above papers focus on vision representation. Other research has directly addressed visual models for mobile robot navigation. Talluri and Aggarwal[37] present a method for ground-based robot self-localisation in outdoor urban man-made landscapes based on straight-line correspondences. They partition free-space into non-overlapping regions over which the visibility list of edges is the same. The system restricts the number of candidate regions using a modified Hough transform and then evaluates each using geometric constraints on possible model-image features. The search space is reduced by an approximate estimate of pose. The system uses a subset of features (roof-top edges) in order to reduce the model size, and assumes that these edges are parallel to the ground plane. Fennema *et. al.*[18] describe a model-based system that matches 2D-line models to features of the environment to obtain its pose and facilitate mobile robot navigation. Both systems identify landmarks to move through the environment rather than to facilitate interaction with objects. These can be seen as similar problems (the robot is inside a large object). However, for navigation in a wholly known environment, Talluri and Aggarwal needed to restrict the number of features in order to control the size of the model. Whereas, for object navigation in an unknown environment, more information is often required to uniquely discriminate the object from the un-modelled background.

Arkin and MacKenzie[2] employ temporal sequencing of a series of navigation algorithms to achieve complex tasks. After identifying the approximate location of an object, a second algorithm is invoked, which uses a constrained Hough transform[38] to identify an object and move close to it based on an object target region of known size and position. To transform the model, the Hough method requires an estimate of relative object location, accurate to within several feet. After moving close to the desired location, a third algorithm handles final docking. Our system could be used for the intermediate step for this type of operation; however, our system does not require transformations or location constraints, and the robot may approach the object from

any angle.

1.2. Benefits of recognising objects for mobile robots

The ability to recognise objects enables robots to perform purposive, deterministic actions on specific objects rather than general exploration and navigation. In order to interact with objects, localisation-based systems require knowledge of object pose and position with respect to the environment. This is not robust behaviour if the object is not always precisely aligned. Systems which recognise objects often use simple 2D recognition (e.g. Ref. [28]). Non-affine rotations of the object will render it unrecognisable in this case. Navigating around objects introduces new problems. Background objects that are similar to the specified object may distract the robot, resulting in it moving to the wrong object, or in the wrong direction. Laser range scanners can provide accurate structural data and segmentation[24], and successful recognition systems based on range data have been developed[19]. However, vision is able to discriminate objects based on surface markings which result in intensity edges, but not in range edges, and thus do not appear in range data.

2. Circumnavigating System

Circumnavigation is the process of navigating completely around an object in a single direction, without colliding, or moving away from the object, unless obstacles or the required tasks make moving away necessary. Our architecture consists of two subsystems running on separate processors which communicate via a serial line by message passing. The circumnavigation subsystem, which guides a robot along the ground plane around 3D objects, is the topic of this paper. The other subsystem[26] controls low-level navigation functions including odometry, and uses a separate camera mounted on the robot to perform vision based obstacle detection.

Figure 1 shows our system architecture. Pre-processing extracts useful features (e.g., edge segments) from a raw image, see Section 3.4 for details. The system recognises a 3D object regardless of pose, assuming it is close enough, matching features to views in a canonical-view model, and finds image-to-model correspondences. The correspondences are used to calculate the four degree of freedom inverse perspective transform, giving approximate object orientation and position from geometric correspondence points that are known for each canonical-view. Geometric boundaries in the model of the object are extended to generate a path around the object. See Ref. [5] for details of the path planning and local navigation. Before making a move, the circumnavigation subsystem polls the obstacle detection/odometry subsystem to check the intended path for obstacles. The pan head locks on the estimated object position as the robot moves. The system checks that each new object view is expected given the robot's belief about object pose, and its own relative position and motion, preventing accumulation of uncertainty inherent in navigation and odometry.

3. Object Recognition

Our system recognises both the object and the view using a viewer-centred, model-based recognition technique, called canonical-views. Viewer-centred representations

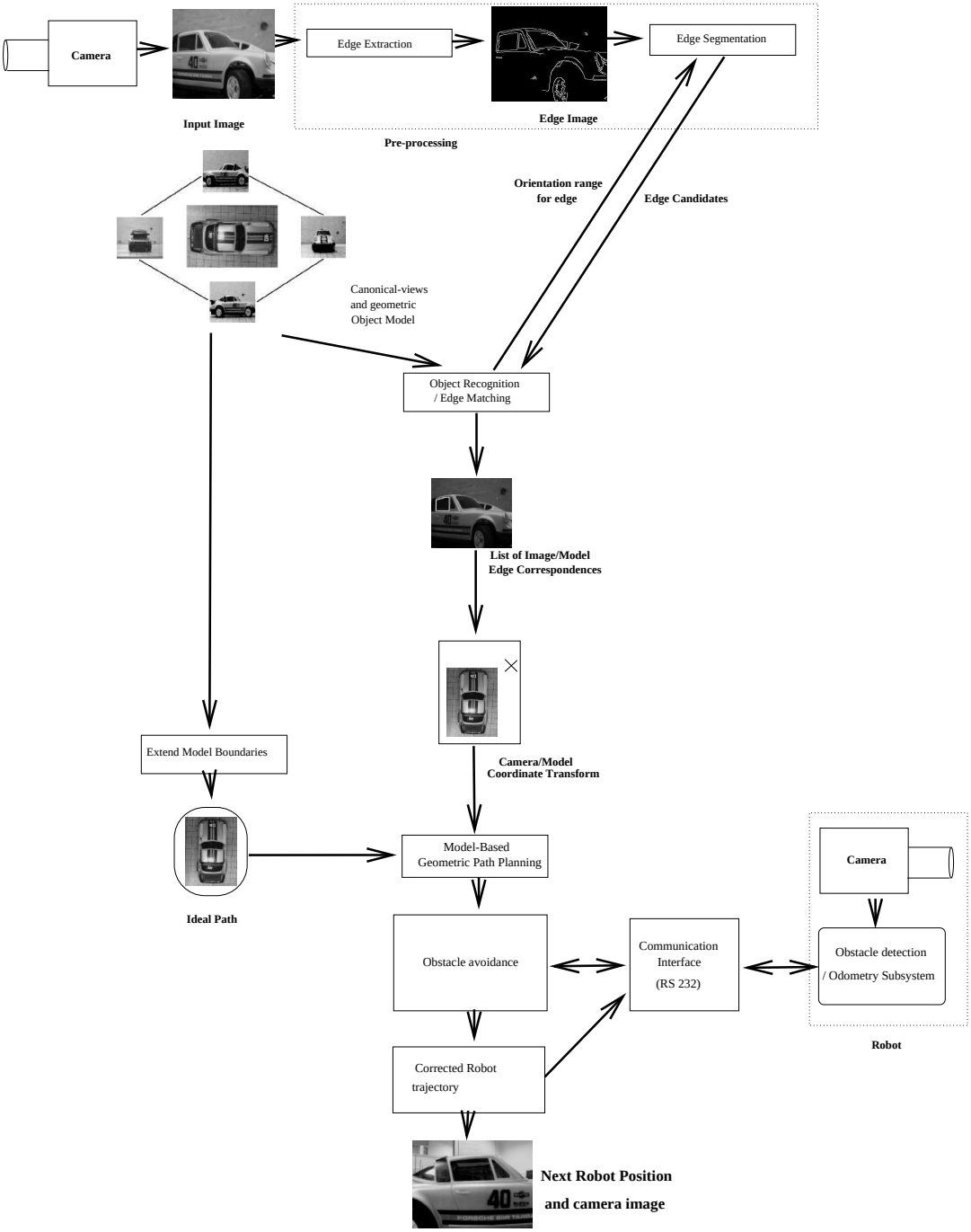


Figure 1: Our system architecture.

generally include a concept similar to a *characteristic view*, which is a continuous region of views, for which all visible edges and vertices, together with their connectivity relationships are the same[39]. Neighbouring characteristic views are distinguished by a change in the visual feature topology of the object. Characteristic views are general in the sense that a single view represents a continuous range of viewpoints around the object.

Viewer-centred representations have been criticised for problems of practicality. For example, aspect graphs[17] may redundantly represent one surface in many views, which are differentiated only by visual events that are not observable in practice. Further, there is no adequate view indexing mechanism. More recently, Eggert *et. al.*[16] presented a thorough analysis of the scale problem for aspect graphs. Other researchers have improved the indexing performance for viewer-centred representations using probabilistic methods[7]) or hierarchical methods[8].

The index and scale problems can be seen as problems of *general* vision. For general vision, every possible detail of an object should be represented, leading to a large representation. However, general vision only exists in theory[1]. In practice, vision is always task-based. The amount of information that must be stored in an object model is thus dependent on the task for which vision is required, as well as the object itself. For example, a vision system to support a robot for spot-welding a vehicle is likely to need more detailed information than if the robot only has to move around the object.

Robot vision can be treated as a very different problem to traditional computer vision, because a robot system is *embodied*.[6] Canonical-views can take advantage of the constraints imposed on recognition by robot navigation. Ground-based robots have a characteristic perspective of objects that defines the scale at which object views can be perceived, and at which features can appear. This perspective is also constrained by the task the robot performs. Further, when a robot has to deal with a particular object, the indexing problem is reduced to a single object. Combining this with the causality of robot motion, and the use of canonical-views, we can generally reduce matching to a single view (other than for drop-off). The canonical-view representation also considers issues of practicality in object modelling:

1. Real cameras are not point size but have finite size.

2. Objects have features that contribute little to recognition.

3. Robots are not generally expected to recognise objects at very close range.

4. Camera images are discrete, thus for a particular viewing range, features below a certain size are not distinguishable.

5. The sequence of views of an object that a robot encounters as it travels around it are *causally* related.

Point 3 requires particular clarification. In order to perform interactions with an object, such as manipulation, a robot must move close to the required surface. Once a robot has identified an object, and moved to be close to the object, facing the required

surface, existing systems are well developed for docking. The robot generally does not need to move closer than tens of centimetres before these mechanisms can be engaged[34]. For a robot performing a set of required interactions with an object, there is a set of likely paths which the robot will take around the object. We call this the *effective view-space* of the object for the robot. The actual scale of features that the robot must recognise is defined by the effective view-space of the object for the robot, and the focal length of the robot's camera. We condense these factors to a single parameter for clarity of definition. Our definition of a canonical-view emphasises the difference between a system for general object recognition, and a system for object recognition specifically to facilitate mobile robot navigation.

Definition 1 *A* canonical-view *is a set of neighbouring characteristic views which: are differentiated only by visual events involving features of size s for which* $\frac{s}{b} \leq k$*, where b is the size of the largest surface in the view, and k is constant; and only models features that contribute significantly to recognition.*

The threshold k can be set in practice by using images of the required object taken by the robot from the range of distances that the system can reasonably be expected to view the object given the task required.

Eliminating redundant small features from the model is important as small features, such as car wheel-nuts, may generate many visual events (see Figure 2(a)). The wheel-nuts may be too small to be recognisable at the distances required by the task. Further, in the case of the wheel-nuts, the extracted edges are poor due to low contrast (see Figure 2(b)).

A canonical-view may also exclude surface texture and features which occur frequently in the background. Object surface markings, such as stripes, that are large with respect to the size of the object can form useful features. Generally, features can be included in the model if they are large enough and have sufficient contrast for them to appear reliably in images from typical operating distances for the required task. Features, which may arise from surface texture or markings, that appear small or do not appear at all from the required distances should be ignored in the model.

We index the representation by the robot path around the object. Thus, canonical-views represent the causal relationship between object views defined by the robot navigation problem described here.

Definition 2 *A* canonical-view representation *defines a link between all views that are neighbouring in the robot's path.*

To reduce the number of model views, we may exclude views where the features are a subset of another view. For example, an aspect graph for three neighbouring faces of a cube (see Figure 3) would have seven views. A possible canonical-view representation would have nodes for views one, three and five. Images with more than one surface visible could match with any of the corresponding canonical-views. However, for simple objects such as cubes, it may be undesirable to reduce the model to this extent. The additional views constrain the possible pose, and further, more features in a view model may aid unique discrimination of the object and view. In general, redundant views can be considered for elimination based on the relative size of their visibility region in effective view space. If the visibility region of a view is smaller

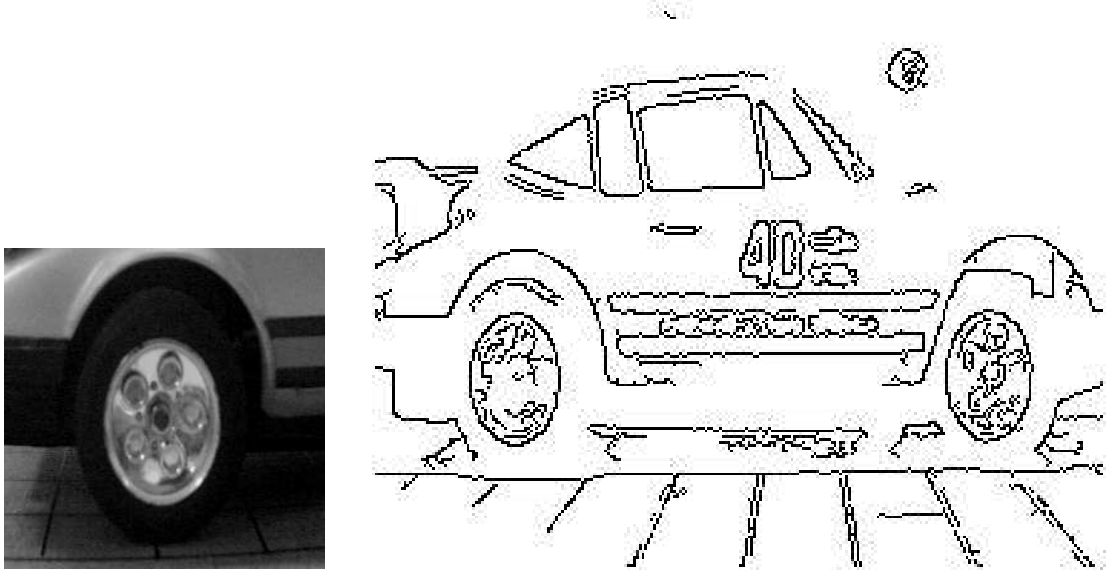


Figure 2: Edges extracted from the wheel nuts are unclear and inconsistent due to poor contrast, and thus make poor features.

than a threshold function of the views of which it is a redundant subset, it may be eliminated.

In certain cases of task-based robot vision, object holes and protrusions may be treated differently than the way they are treated in general computer vision. If the robot is of similar scale to the object, or larger, its physical body may often prevent it from moving into close proximity to holes or protrusions. In such cases, a hole or protrusion can be treated as a standard surface feature, and thus views of the hole or protrusion that are only visible from very close to the object may be excluded from the model. However, if an object has holes or protrusions that are large enough that the robot may move within the object's convex hull, and the task required of the robot may make such action necessary (e.g., docking), the model may need to include additional views based on images from such positions.

For a particular view match, segmentation errors may lead to a mismatch to a similar view. However, as a robot moves around an object it can combine evidence from several *causally* related views, greatly increasing the certainty of identification.

Figure 4 shows the views and their relations that were modelled for a model car. The view models consist of a list of lines that must appear. The lines have: a unary feature of the angle at which the line must appear; binary features of basic spatial relations to other edges; and, 3D object centred coordinates of the end-points to facilitate geometric verification and pose determination. Model feature value ranges in our experiments were derived manually from segmented edge images of the views, with variance estimated using the perspective equations. To derive practical models for robot navigation

it is important to consider what the robot can actually perceive, rather than geometric derivation from a CAD model.

Note that the number of views that are required to model an object and the detail of features in each view is dependent on the object itself, but also on the task required of the robot.

3.1. Features

Burns *et al.*[9], prove that there are no general-case invariant visual features under perspective projection, as features values "blow out" for some views, see Figure 5(a). However, some features exhibit *low view variation*, where the variation is small in extent over a large fraction of the views. In Figure 5(a), although the features of surface 1 have extreme values, recognition is still possible based on surfaces 2 and 3. In order to restrict the initial list of candidate matches in our canonical-view model we eliminate candidates based on simple 2D features. These are: edge orientation (Continuous range); and relative edge location ([Above, Below], [Left-of, Right-of]). We subsequently use geometric verification and analyse how much of the model is covered by the extracted image edges to choose the best match.

We use edge orientation, η, relative to the image y-axis as there is only rotation about Z, see Figure 6. Perspective variance of η is proportional to the angle, α, of the edge to the image plane, and the relative displacement, Δd, of the line end-points from the camera focal centre. Also, as α increases, foreshortening of the horizontal component of the edge, η may decrease. Edge orientation ranges can be estimated using Equation (8) from the next section at the extreme points of required view visibility, or empirically found from images at extreme views.

Relative edge location is a useful feature due to restricted robot motion. An edge pair is labelled if every point on one edge is above/below/left-of/right-of every point of the other, for every viewing angle. If a pair of edges has a left-of or right-of relation at one point of view space, it will be true whenever both edges are visible, except when one edge is on a limb which is above or below the surface of the other edge, see Figure 7. Above/Below relations are affected by perspective. Parallel lines, coplanar in Z, will not cross, but, non-parallel edges, or edges at varying depths may, (Figure 8). Perspective effects vary with displacement from the image centre in x, and with α. A second set of binary features: Above1/Below1/Left-of1/Right-of1 denote that at least one point on a particular edge always forms the specified spatial relation with at least one point on the other edge. These are for edges that join.

3.2. Edge matching

The edge matching method used has similarities to that of Fennema *et. al.*,[18] but incorporates more targeted edge processing following the approach of the fast-line finder,[27] and fuses information from odometry for matching. It is only described briefly here, full details are presented in Ref. [3].

The edge extraction method finds edges for a specified range of orientation. We do not consider any edges that are not within the required range for a model edge, where each edge has an independently restricted orientation range. This feature of our edge

extraction method eliminates the vast majority of possible candidate matches without any computation, as the range for each candidate edge is almost always substantially less than $\frac{\pi}{4}$. From the list of candidates, we eliminate all edges that do not satisfy relative location constraints, yielding a vastly smaller set of candidate matches.

3.2.1. Evaluating edge candidates

Matching evaluation is only performed on lines in the model that intersect other lines at their end-points. Not all lines in the model are required to intersect with others, only those that are geometrically evaluated. This is to facilitate matching when only partial edges are successfully recovered. Each match candidate that satisfies the basic spatial requirements is then evaluated.

We employ two criteria for evaluating matches:

- ratio of coverage; and,
- geometric verification.

The ratio of coverage is the amount of the hypothetical true edge that the extracted edge covers, based on an estimate of the intersection of the infinite lines defined by the extracted image edges according to a matching hypothesis. Figure 9 shows a hypothetical match. We have three sets of edges, the observed image edges e^i, the extended edges which terminate at the intersections, e^e, and the model edges, e^m. We form a set of $e^{i'}$, the part of each e^i that overlaps the corresponding e^e. For each $e^{i'}$ we form the coverage ratio:

$$e^r = \frac{||e^{i'}||}{||e^e||} \tag{1}$$

To evaluate the match we take an average of the ratios for all edges matched, c^m. In order to allow for one poorly extracted edge we may ignore the edge with the minimum e^r:

$$c^m = \frac{\sum_{i=0}^{n} e_i^r - min(e_i^r)}{n-1}, \tag{2}$$

where obviously from Equation (1), $0 < c^m \leq 1$. Candidate matches that generate values of c^m that are less than a threshold, c^t are eliminated.

3.3. Geometric verification

Geometric verification of matches for the general case is discussed by Grimson[20]. This is a special case due to the restricted rotation relation discussed previously. Geometric verification is performed by calculating the pose associated with a candidate match, and checking that the pose is consistent with the robot motion, and that all points in the candidate match are consistent with the pose.

We calculate the pose estimate associated with the points of intersection for the match. This is compared with the estimate of object pose from the odometric reading of the robot, based on the previous estimation of the object position and the robot motion

between the two images. The match is rejected if the difference between the distance to the object predicted from the current match, and that predicted by odometry is greater than a threshold. Similarly, for the estimated object orientation.

Evaluation of the geometric consistency of the match is implemented by projecting the model edges into a virtual image, based on the pose estimate. The similarity between the virtual image and the real image is then measured by the total least squared differences in the length of the normalised edges, and on the sum of the absolute difference of the angles. The match is rejected if the difference is larger than a threshold.

We form a measure of geometric consistency, g^m, by a linear combination of the geometric measures. To combine c^m and g^m, we subtract the threshold, c^t, from c^m, and normalise the result to increase the discrimination between candidates based on coverage, without increasing its emphasis overall. This is added to the normalised result of the combination of the geometric measures, to form the evaluation of the match e^m:

$$e^m = \frac{c^m - c^t}{1 - c^t} + g^m. \tag{3}$$

We find e^m for any matches that have not been rejected, and take the minimum:

$$m^s = MIN_{i=0}^{p} e^m. \tag{4}$$

Full details of the matching process are available in Ref. [3].

3.4. Edge Extraction

Edge extraction in this system is designed to be fast, but also to produce edges that are adequate to match reliably. Only straight line edges are extracted.

We use the Canny edge extractor[10], followed by dual threshold edge synthesis, which gives a good quality starting edge image. We apply a linear fit of n x n windows to the edges over the entire edge image, discarding any windows with less than $\frac{n}{k}$ edge pixels present. These edge windows are coarsely grouped in bins in ρ, θ space.

At this point pre-processing is complete, and the majority of the n x n windows will not be considered again. We now expand all window bins that fall within the required range of θ for the model edges, as requested by the matching subsystem. This expansion joins edge windows within the bin by examining all windows in between and checking if they can be validly included as part of an edge. Zero or more actual edges will be found within each range.

Full details of the edge extraction process are available in Ref. [3].

4. Determining Object Pose and Distance

Typical methods for finding pose use numerical methods and allow six degrees of freedom [32]. However, our robot moves on the ground, and camera elevation can be varied without calibration. As a result there are four degrees of freedom (one rotational and three translational) for calculating the respective locations of the robot and object. Further, general robot navigation does not require precise pose, so we can derive a fast,

algebraic estimate. Techniques that assume precise knowledge of camera height may be sensitive to camera tilt and ground plane irregularities.

Figure 10 shows the model / camera coordinate transform. By inspection the pinhole camera perspective equations for image coordinates u and v are:

$$u = \frac{f(T_x + Y_m \cos\theta - X_m \sin\theta)}{T_y - Y_m \sin\theta - X_m \cos\theta}, \tag{5}$$

$$v = \frac{f(Z_m + T_z)}{T_y - Y_m \sin\theta - X_m \cos\theta}, \tag{6}$$

where X_m, Y_m, Z_m are the model coordinates, and T_x, T_y, T_z are the translations along the camera x, y and z axes of the model coordinate system.

Dividing (6) by (5) gives:

$$\frac{v}{u} = \frac{Z_m + T_z}{T_x + Y_m \cos\theta - X_m \sin\theta}. \tag{7}$$

Using edge segments rather than points eliminates translations (T_x, T_y, T_z):

$$\frac{\Delta v}{\Delta u} = \frac{\Delta Z_m}{\Delta Y_m \cos\theta - \Delta X_m \sin\theta}, \tag{8}$$

where $\Delta u = u_2 - u_1$, $\Delta v = v_2 - v_1$, $\Delta X_m = X_{m2} - X_{m1}$ $\Delta Y_m = Y_{m2} - Y_{m1}$, and $\Delta Z_m = Z_{m2} - Z_{m1}$, and (u_1, v_1) and (u_2, v_2) are two points in the image, and (X_{m1}, Y_{m1}, Z_{m1}) and (X_{m2}, Y_{m2}, Z_{m2}) are corresponding model coordinates.

We may now deduce the required equations. Substituting $Y\cos\theta - X\sin\theta = \cos(\theta + \alpha)\sqrt{X^2 + Y^2}$, where $\tan\alpha = \frac{Y}{X}$ into (8) gives (9). We derive (10) by considering (6) for the vertical component of an edge, where $\overline{X}_m = (X_{m1} + X_{m2})/2$, and $\overline{Y}_m = (Y_{m1} + Y_{m2})/2$ are the mean model coordinates of the edge, and $\overline{T}_y = (T_{y1} + T_{y2})/2$ is the mean of the translations in Y to model points (X_{m1}, Y_{m1}, Z_{m1}) and (X_{m2}, Y_{m2}, Z_{m2}). Finally, consider (5) for the displacement of the vertical edge used in (10) from the image centre, where $\overline{T}_x = (T_{x1} + T_{x2})/2$ is the mean displacement of the model points.

$$\cos(\theta + \alpha) = \frac{\Delta Z_m \Delta u}{\Delta v \sqrt{X^2 + Y^2}}, \tag{9}$$

$$\overline{T}_y = \frac{f \Delta Z_m}{\Delta v} + \overline{Y}_m \sin\theta + \overline{X}_m \cos\theta, \tag{10}$$

$$\overline{T}_x = -u \frac{\overline{T}_y - \overline{Y}_m \sin\theta - \overline{X}_m \cos\theta}{f} + \overline{Y}_m \cos\theta - \overline{X}_m \sin\theta. \tag{11}$$

This derivation could be modified to include camera tilt for known angles.

Equation (9) gives only magnitude of θ. We determine the sign by finding the closer of two edges based on relative ratios of image to model displacement in v. This assumes the object is close enough for perspective projection.

θ and its sign are sensitive to edge length for θ close to zero. We set θ to zero if a p pixel change in edge length would reverse our sign estimate. In practice, this leads to approximately a 20° uncertainty in estimates of θ, for θ close to zero. This is acceptable for navigating around an object at a distance that is large with respect to the object

size, but may often be inadequate for close interaction. In such cases, the ambiguity is greatly reduced by fusing orientation estimations from several views.

4.1. Model-Based Path Planning

Circumnavigation can be performed as a series of local navigation problems. At each point the robot assesses its relative position, and moves along the object surface, remaining a minimum safe distance from the object. To ensure that the robot does not collide with the object, we require that the robot move only a short distance in comparison to its total distance away from the nearest point on the object before it visually checks its position. The distance the robot should remain from the object is task dependent. For instance, to dock, a robot must move close to a particular point after navigating around the object.

We derive an *ideal* path around the object by extending object boundaries in two dimensions by the safe distance, similar to the *configuration space* approach[33], see Ref. [5] for details. Other approaches to path planning may produce a smoother path, however, this is not the focus of the paper. The path is in object-centred coordinates and considers only the object. To use it in practise, it must be transformed into current robot coordinates using the object pose found through recognition, and then into odometric coordinates to control the odometry subsystem. The robot needs only to follow the path approximately and must handle any obstacles using local navigation strategies.

5. Experimental Results

We present trials that demonstrate our system. Firstly, the object recognition and pose determination is evaluated in isolation. Then, three navigation trials are presented to demonstrate the performance of the entire system. The experimental setup used in the trials with the power supply is shown in Figure 11, trials for the model car shown in Figure 16 used a similar configuration.

5.1. Evaluation of recognition and pose determination

To evaluate the vision system, images were taken at a distance of 1550 *mm* from the object. The object was rotated on a turntable while the robot remained fixed. A set of five images was taken from 18 angles at intervals of 20° around the object. The distances reported are the total distance from the centre of the robot to the centre of the object. The images within the data set were taken against the same background, which has some distracting features. A sample image is shown in Figure 12.

Figure 13 shows a scatter-plot of the distances predicted by the matching and pose determination against the angle from which the image was taken. In this trial, the robot was given an accurate *a priori* estimate of object pose. The scatter-plot shows that 63% of estimates fell within 50 *mm* of the true value. Only 7% of the estimates fell outside 100 *mm*. These 7% correspond to cases where the system has partially mismatched the object. Figure 14 shows a scatter-plot of the system's angle estimates. The errors in the Y offset can be large in comparison with the actual value. However, the results for the Y offset were not as important, as it is always small while the object

is in the image. The Y offset is important for object fixation, but this has not been directly addressed in this paper.

For images with no background features, matching results improved marginally. With more background features, the reliability of matching is not greatly reduced, however, more candidate matches must be evaluated so computation time can increase.

Images of the object were also taken at larger and smaller distances. We found that if the object was entirely visible in the image, the results were similar for smaller robot-to-object distances. The system performance began to degrade substantially at a robot-to-object distance of 2150 *mm*. As the edges become small in the image, both matching and pose determination accuracy can be expected to deteriorate.

In trials where the system was given an incorrect *a priori* estimate of object pose, the output pose estimates remained largely stable, up to the point where the error in the *a priori* estimate was large enough that the correct value would be penalised. Figure 15 shows the graph for the same data set, when the *a priori* distance estimate was 1700 *mm* (actual distance was 1550 *mm*). The estimates of orientation remained stable also, except for the case where the incorrect angle estimate lead to an incorrect estimation of the view. This problem could be alleviated somewhat by attempting to match to more than one candidate view when the match is on a view boundary.

In the case where object pose is not known *a priori*, but the view of the object that the robot is facing is known, the system was able to match successfully only 46% of the time for the power supply object. For the model car, the results for the drop-off problem were much better, with more than 80% of distance estimates falling within +/- 150 *mm*, when only the view was specified. This difference arises from the fact the modelled features of the power supply consist entirely of vertical and horizontal lines. Such features frequently appear in indoor office environments, so mismatches are likely without fusion of odometric information. However, the model car incorporates some angled edges that occur far less frequently, thus fusion of odometric information was less critical.

Further, for the power supply there were no features that strongly distinguish one view from another, so for many viewpoints the robot matches any view to an image with similar certainty. However, for the model car, unique identification of the view is possible. For a set of four images, one taken from each side of the car, the correct view was ranked highest by the system in two cases, and second highest in the other two.

This data implies that a naive approach to the drop-off problem would not be reliable. A more sophisticated start-up procedure utilising multiple consistent matches before making large moves may be necessary.

5.2. Circumnavigation trials

In the first full system trial, the robot moves around the corner of the model vehicle. The robot recognises the object surface of the model vehicle that is clearly visible. The robot moves based on this recognition and then identifies the next view as it becomes visible, while the current view disappears. Simple tracking-based systems are unable to handle moving around corners in this manner. Moving around corners presents a difficult problem for tracking-based systems, as without a model, what is the object

and what is the background is not well-defined.

Figure 16 shows the images taken by the robot for recognition. The wind-screen and rear window make up the model for the front and rear of the car. The rear and middle windows were modelled for the side surfaces. The matches are overlayed on the images in Figure 16, but only the left-most window matched is shown for the side surface in Figure 16 (a), (b), and (c).

In the second experiment the robot demonstrates circumnavigation of the power supply. The system guides the robot around the object, and finally returns to its starting position. Figure 17 shows a graph of the robot moving around the object. This graph is to scale, accurate to within 100 *mm*. Figure 18 shows the images from which the robot recognised the object for the first 10 positions. These are overlayed with the matches for these positions. The robot clearly mismatched the object in frames (e), (g), and (i), with some slight mismatches in other frames. However, these mismatches did not cause the robot to lose the object because the system position estimates were close to the true value. Any matches that lead to pose estimates that were far from the previous pose estimate were rejected. During this trial the object fell narrowly outside the field of view of the camera on several occasions, this was corrected manually. Gaze control is a difficult problem in its own right, and has not been addressed directly in this paper.

Figure 19 shows our robot encountering an obstacle in the third experiment. In this case, it applies an obstacle avoidance strategy of backing off and proceeding to the destination surface in the other direction.

6. Conclusion

We have implemented a vision system to guide an autonomous robot to circumnavigate objects of arbitrary pose. By applying 3D object/view recognition and pose determination the robot is able to navigate safely around a known object. The system visually checks its position frequently to prevent accumulation of odometry errors. The results demonstrate our system can successfully circumnavigate real objects and handle obstacles.

Further research is needed to make the system's performance more reliable. We are engaged in further research to extend our use of active vision: dynamic information will make pose estimation more robust; and tracking of the object more reliable.

1. Y. Aloimonos, "Introduction: Active vision revisited", in *Active Perception*, Lawrance Erlbaum Assoc., Hillsdale, NJ, 1993, pp. 1–18.
2. R. C. Arkin and D. MacKenzie, "Temporal coordination of perceptual algorithms for mobile robot navigation", *IEEE Trans on Robotics and Automation*, **10**, 3 (1994) 276–286.
3. N. Barnes. *Vision-Guided Circumnavigating Autonomous Mobile Robots.* PhD thesis, Department of Computer Science and Software Engineering, University of Melbourne 1999.
4. N. M. Barnes and Z. Q. Liu, "Vision guided circumnavigating autonomous robots", in *ICSC '95 - Proc. Third International Computer Science Conference: Image Analysis Applications and Computer Graphics, Hong Kong*, pp. 33–42 1995.
5. N. M. Barnes and Z. Q. Liu, "Collision-free path planning based on visual information: An application in circumnavigation", in *Fourth International Symposium on Signal Processing and its Applications, Robotics and Robot Vision Workshop*, pp. 50–55 1996.

6. N. M. Barnes and Z. Q. Liu, "Embodied computer vision for mobile robots", in *ICIPS '97 - IEEE Int. Conf. on Intelligent Signal Processing*, pp. 1395–1399 1997.
7. J. Ben-Arie, "The probabilistic peaking effect of viewed angles and distances with application to 3-d object recognition", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **12**, 8 (1990) 760–774.
8. J. B. Burns and E. M. Riseman, "Matching complex images to multiple 3d objects using vi ew description networks", in *IEEE Computer Society Conf. on Computer Vision and Patt ern Recognition*, pp. 328–334 1992.
9. J. B. Burns, R. S. Weiss, and E. M. Riseman, "View variation of point-set and line-segment features", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **15**, 1 (1993) 51–68.
10. J. F. Canny. "Finding edges and lines in images". Master's thesis, MIT Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Massachusetts 1983.
11. I. J. Cox, "Blanche - an experiment in guidance and navigation of an autonomous robot vehicle", *IEEE Trans. on Robotics and Automation*, **7**, 2 (1991) 193–204.
12. S. J. Dickinson, H. I. Christensen, J. K. Tsotsos, and G. Olofsson, "Active object recognition integrating attention and viewpoint control", *Computer Vision, Graphics, and Image Understanding*, **67**, 3 (1997) 239–260.
13. S. J. Dickinson and D. Metaxas, "Integrating qualitative and quantitive shape recovery", *International Journal of Computer Vision*, **13**, 3 (1994) 311–330.
14. S. J. Dickinson, A. P. Pentland, and A. Rosenfeld, "From volumes to views: An approach to 3-d object recognition", *Computer Vision, Graphics, and Image Processing*, **55**, 2 (1992) 130–154.
15. S. J. Dickinson, S. Stevenson, E. Amdur, J. Tsotsos, and L. Olsson, "Integrating task-directed planning with reactive object recognition", in *Proceedings SPIE:Intelligent Robots and Computer Vision X11:Algorithms and Techniques*, pp. 212–224 1993.
16. D. W. Eggert, K. W. Bowyer, C. R. Dyer, H. I. Christensen, and D. B. Goldof, "The scale space aspect graph", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **15**, 11 (1993) 1114–1130.
17. O. Faugeras, J. Mundy, N. Ahuja, C. Dyer, A. Pentland, R. Jain, and K. Ikeuchi, "Why aspect graphs are not (yet) practical for computer vision", in *Workshop on Directions in Automated CAD-Based Vision*, pp. 97–104 1991.
18. C. Fennema, A. Hanson, E. Riseman, J. R. Beveridge, and R. Kumar, "Model-directed mobile robot navigation", *IEEE Trans. on Systems, Man, and Cybernetics*, **20**, 6 (1990) 1352–1369.
19. R. B. Fisher, A. F. ad M Waite, M. J. L. Orr, and E. Trucco, "Recognition of complex 3-d objects from range data", in *Proc. 7th Int. Conf. on Image Analysis and Processing: Progress in Image Analysis and Image Processing III*, pp. 599–606 1994.
20. W. E. L. Grimson, *Object Recognition By Computer: The Role of Geometric Constraints*, MIT Press, Cambridge, Mass., 1990.
21. E. Grosso, G. Metta, A. Oddera, and G. Sandini, "Robust visual servoing in 3-d reaching tasks", *IEEE Trans. on Robotics and Automation*, **12**, 5 (1996) 651–670.
22. G. D. Hager, "The x-vision system: A general purpose substrate for real-time vision-based robotics", in *Proc. Workshop on Vision For Robots*, pp. 56–63 1995.
23. G. D. Hager and P. N. Belhumer, "Efficient region tracking with parametric models of geometry and illumination", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **20**, 10 (1998) 1025–1039.
24. A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. Goldgof, K. Bowyer, D. Eggert, A. Fitzgibbon, and R. Fisher, "An experimental comparison of range segmentation algorithms", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **18**, 7 (1996) 15–26.
25. A. Hormann and U. Rembold, "Development of an advanced robot for autonomous assembly", in *IEEE Int. Conf. on Robotics and Automation*, pp. 2452–7 1991.
26. A. Howard and L. Kitchen, "Fast visual mapping for mobile robot navigation", in *ICIPS '97 - IEEE Int. Conf. on Intelligent Signal Processing*, New Jersey, USA, IEEE, Proceedings of the 1997 IEEE International Conference on Intelligent Processing Systems, pp. 1251–1255 1997.
27. P. Kahn, L. Kitchen, and E. M. Riseman, "A fast line finder for vision-guided robot navigation",

IEEE Trans. on Pattern Analysis and Machine Intelligence, **12**, 11 (1990) 1098–1102.
28. D. Kappey, J. Raczkowsky, U. Rembold, and R. Dillmann, "Knowledge-based object recognition with uncertainty handling mechanisms", in *Proceedings of an International Conference. Intelligent Autonomous Systems 2. IOS*, **1**, pp. 393–9 1990.
29. K. N. Kutulakos and C. R. Dyer, "Recovering shape by purposive viewpoint adjustment", *International Journal of Computer Vision*, **12**, 2-3 (1994) 113–126.
30. J. L. Leonard and H. F. Durrant-Whyte, "Mobile robot localizations by tracking geometric beacons", *IEEE Trans. on Robotics and Automation*, **7**, 3 (1991) 376–382.
31. T. C. Leuth, U. M. Nassal, and U. Rembold, "Reliability and integrated capabilities of locomotion and manipulation for autonomous robot assembly", *Robotics and Automous Systems*, **14** (1995) 185–198.
32. D. G. Lowe, "Three-dimensional object recognition from single two-dimensional images", *Artificial Intelligence*, **31**, 3 (1987) 355–395.
33. T. Lozano-Perez, "Spatial planning: A configuration space approach", *IEEE Trans. on Computers*, **C-32**, 2 (1983) 108–120.
34. K. Mandel and N. A. Duffie, "On-line compensation of mobile robot docking errors", *IEEE Int. Journal of Robotics and Automation*, **RA-3**, 6 (1987) 591–598.
35. J. Santos-Victor and G. Sandini, "Visual behaviours for docking", *Computer Vision and Image Understanding*, **67**, 3 (1997) 223–28.
36. K. T. Sutherland and W. B. Thompson, "Localizing in unstructured environments: Dealing with errors", *IEEE Trans. on Robotics and Automation*, **10**, 6 (1994) 740–754.
37. R. Talluri and J. K. Aggarwal, "Mobile robot self-localisation using model-image feature correspondence", *IEEE Trans. on Robotics and Automation*, **12**, 1 (1996) 63–77.
38. D. L. Vaughn and R. C. Arkin, "Workstation recognition using a constrained edge-based hough transform for mobile robot navigation", in *Proceedings SPIE:Sensor Fusion III:3D Perception and Recognition*, pp. 503–514 1990.
39. N. A. Watts, "Calculating the principle views of a polyhedron", in *9th International Conference on Pattern Recognition*, pp. 316–322 1988.

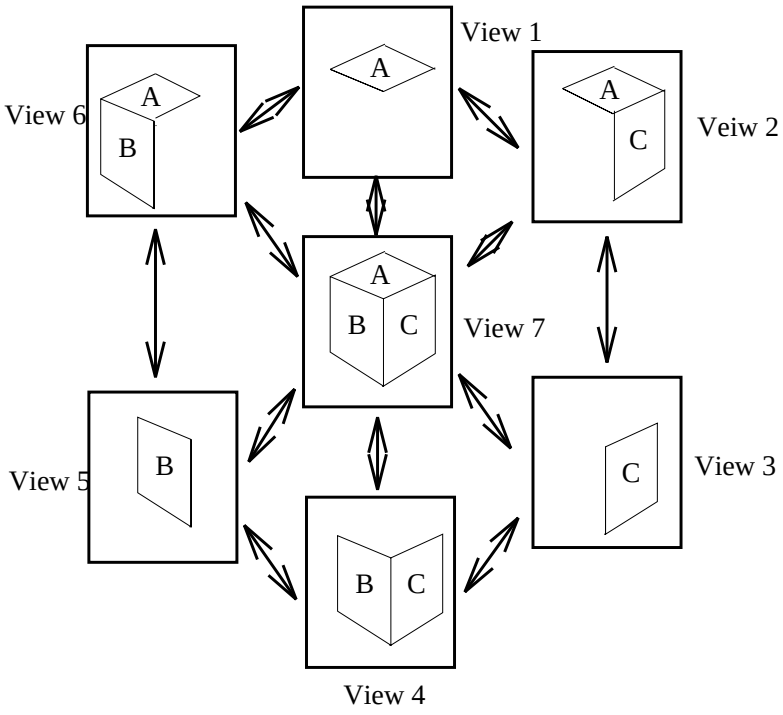


Figure 3: A partial aspect graph representation of a cube. Views 1-6 all consist only of subsets of the surfaces appearing in view 7.

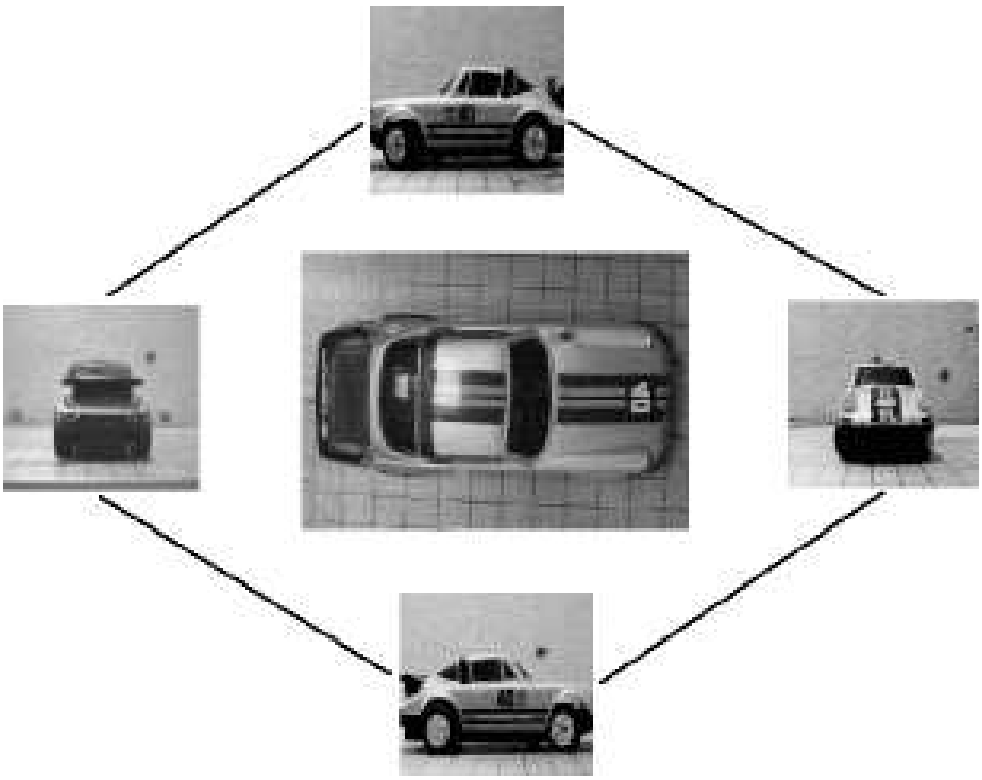


Figure 4: The four canonical views of a model car. Each view represents the adjacent surface of the car. The views overlap at the car corners so that small odometry/pose errors will not lead to incorrect view estimation.

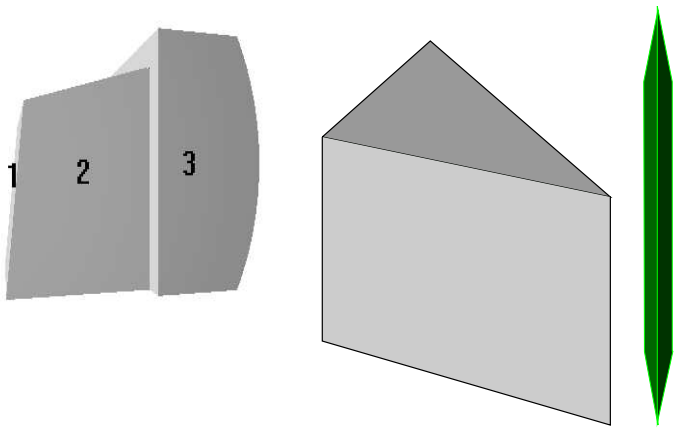


Figure 5: Examples of feature "blow out": (a) The features of surface 1, the back of a computer terminal, are at extreme values, surface 2 and 3 are normal, (b) a wedge shaped object, (c) view from the tip of the wedge, all visible surfaces are distorted, such objects are difficult for viewer-centred object recognition.

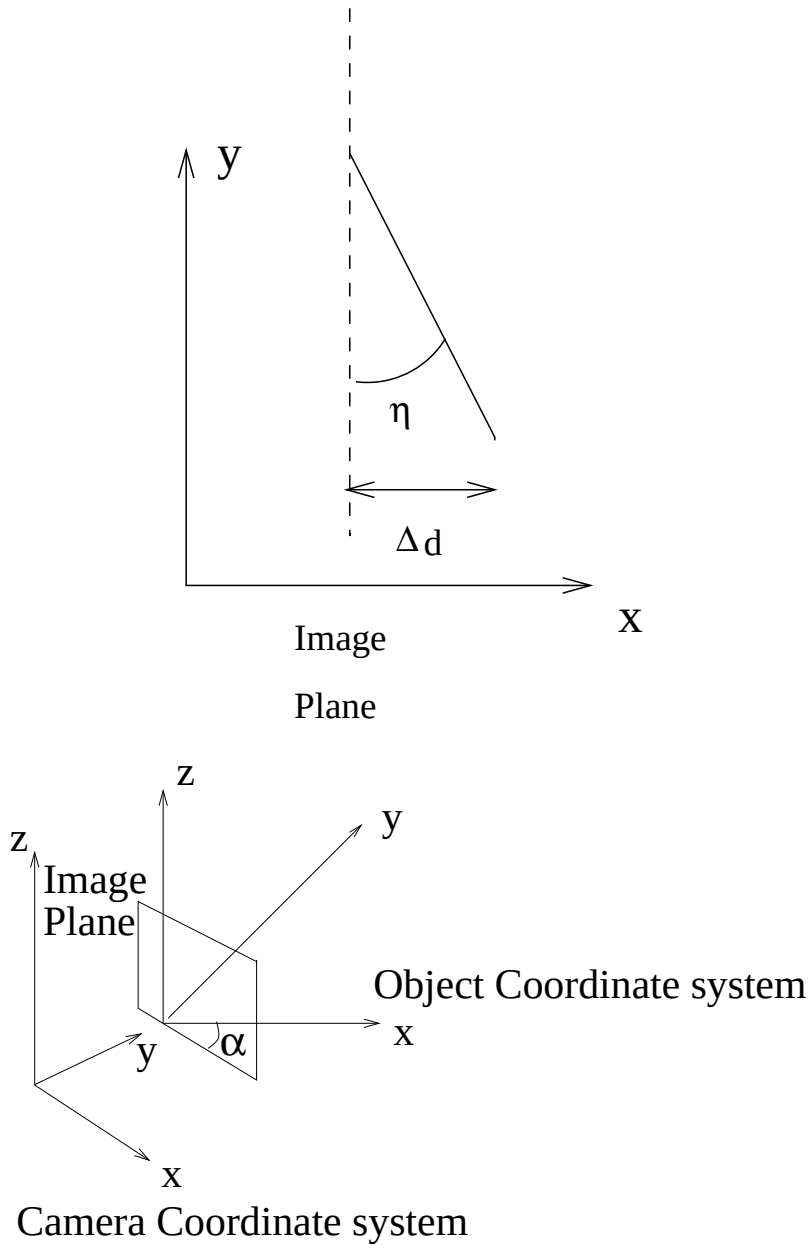


Figure 6: Change in orientation of edges, η, is restricted as the image plane only rotates with respect to the z-axis.

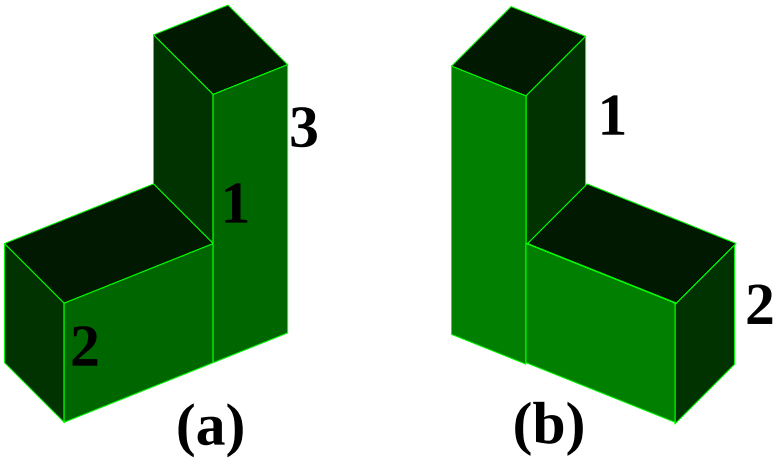


Figure 7: In (a) edge 1 is right-of edge 2, in (b) edge 1 is left-of edge 2, but edge 1 is left-of edge 3 whenever both are visible.

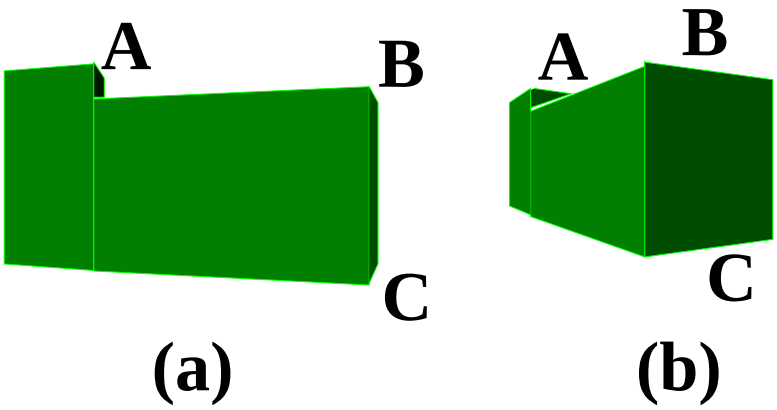


Figure 8: Edge B and C are parallel and at the same depth, however, edge A is at a different depth: (a) A is above B, B is above C, (b) B is above C, but A is not above B.

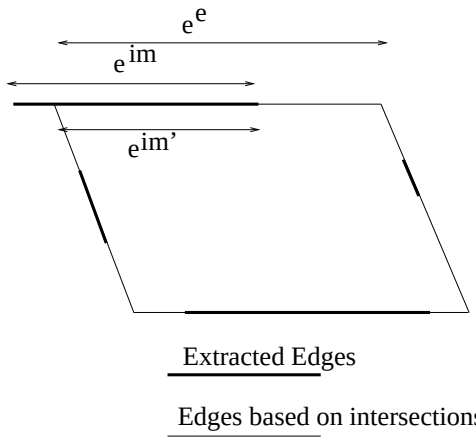


Figure 9: Based on the candidate edges and the model, intersections are calculated to find the match for geometric evaluation. There are three lengths for each of the edges: e^i, the edge extracted from the image; e^e, the edge that terminates at its intersections with connecting edges from the model; and $e^{i'}$, the part of e^i that overlaps e^e. The coverage ratio is $\frac{e^{i'}}{e^e}$.

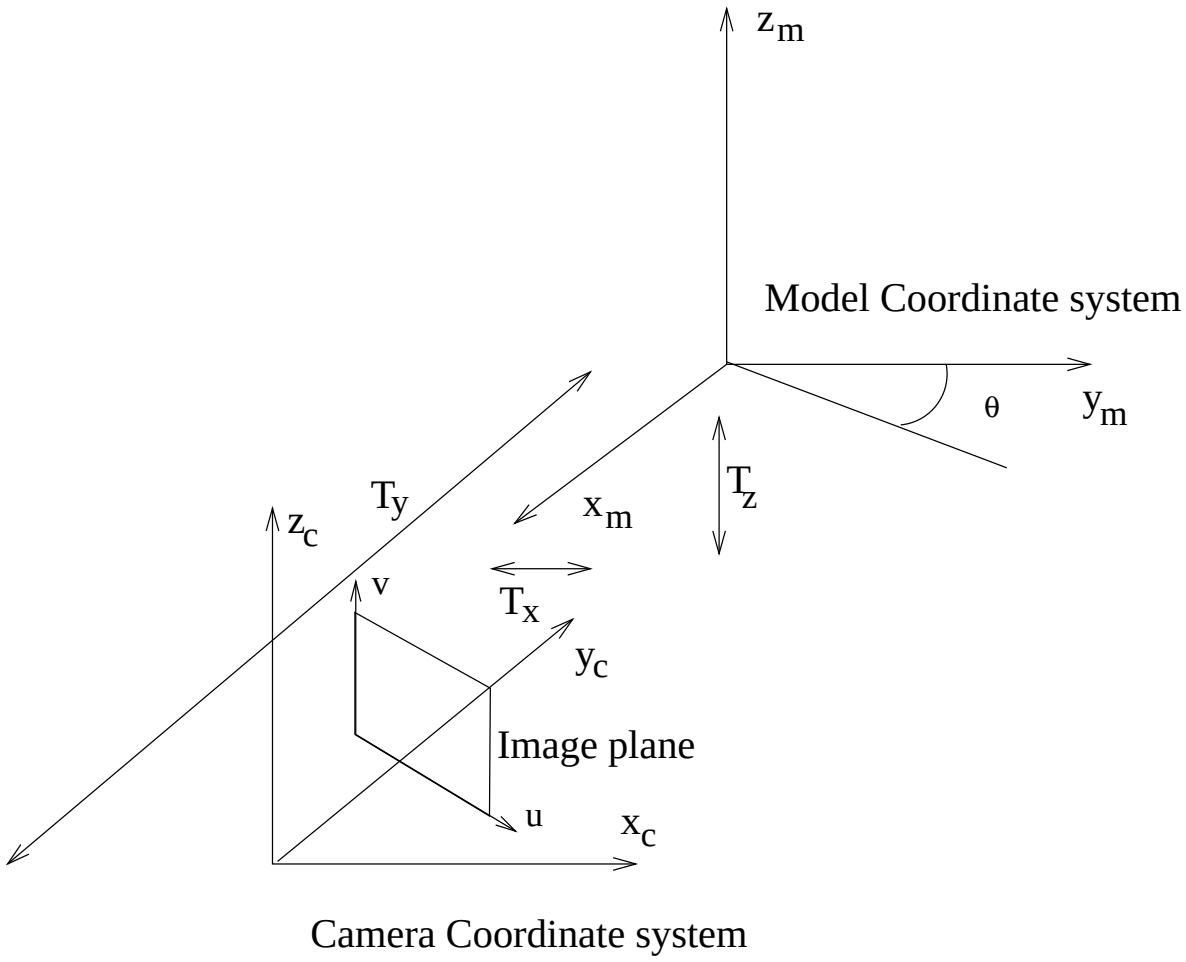


Figure 10: The transformation from the model to the camera coordinate systems.

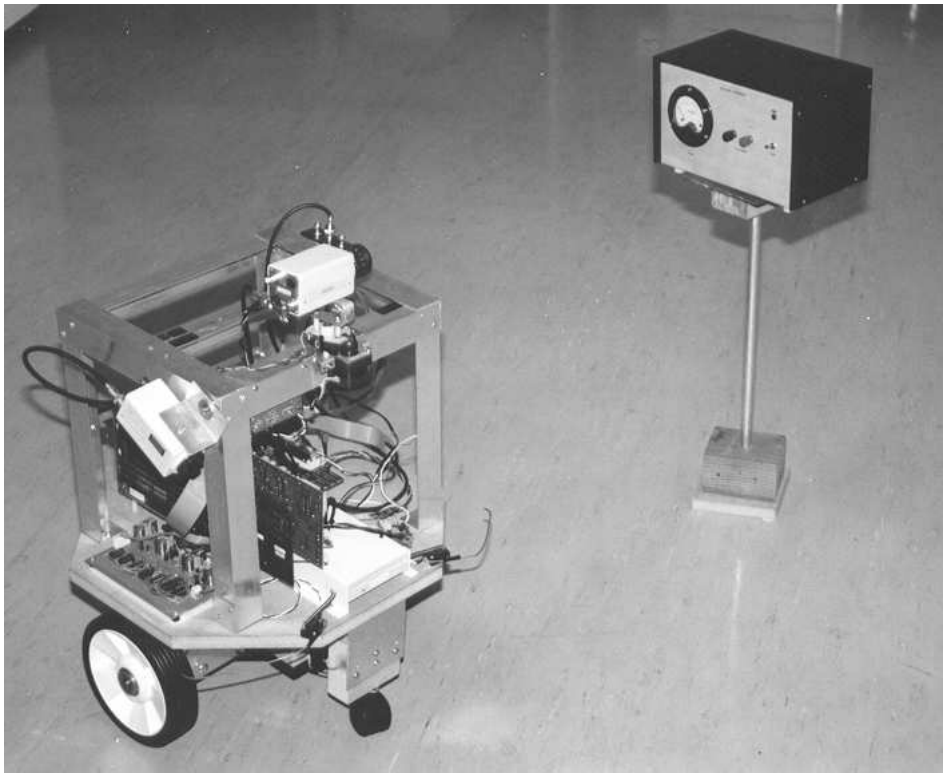


Figure 11: The robot navigating around the power supply in the Computer Vision and Machine Intelligence Lab (CVMIL) at the University of Melbourne. The robot is equipped with two cameras, one fixed and pointing forward for obstacle detection, and one on a pan-head for recognising and tracking the object.

Figure 12: One of the test images of the power supply, overlayed with the match to it.

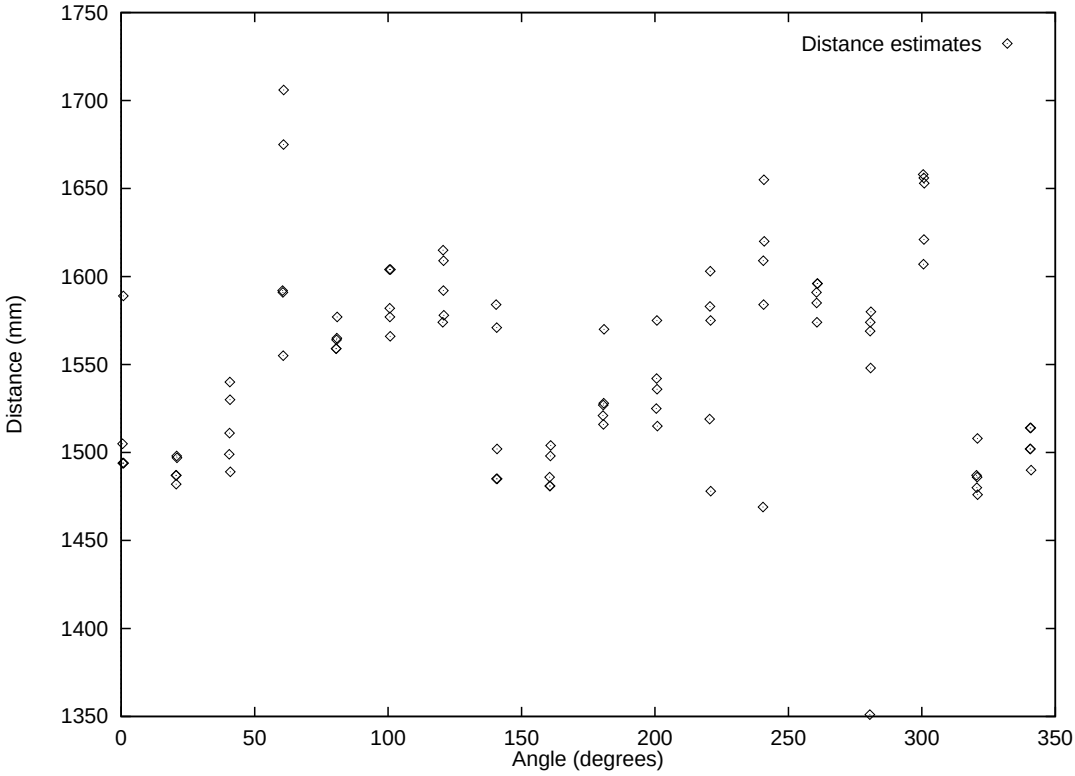


Figure 13: Scatter plot of the distances predicted by pose determination against the angle from which the image was taken.

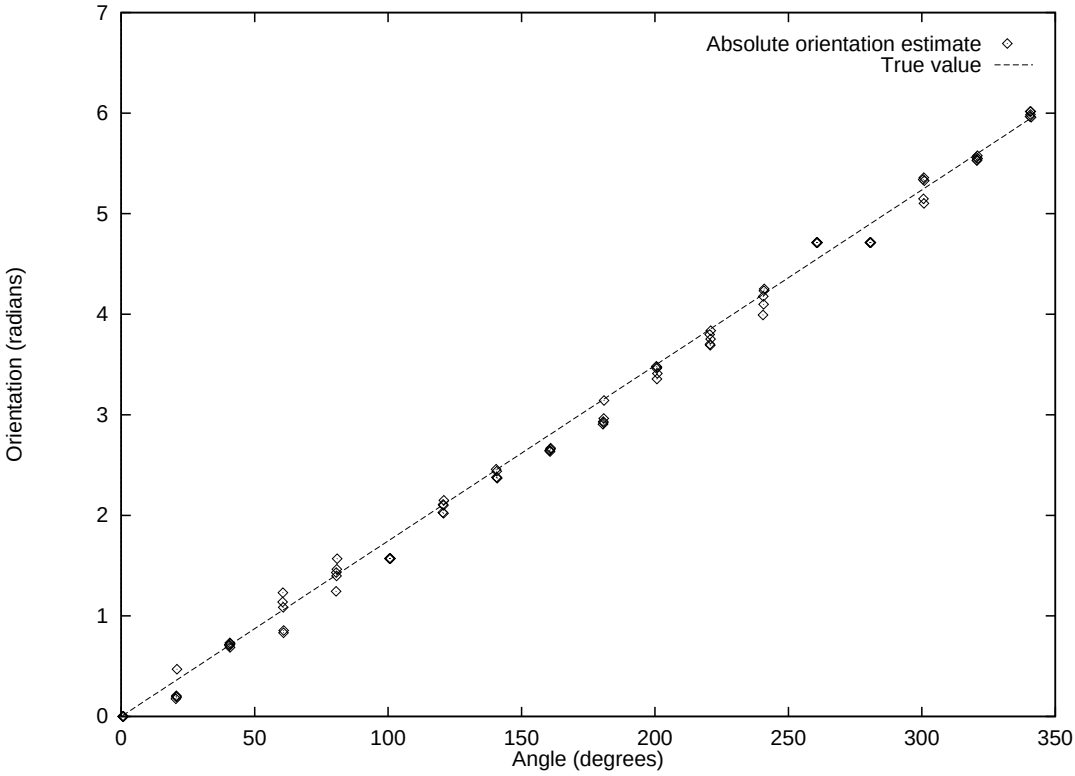


Figure 14: Scatter plot of the angle predicted by pose determination against the angle from which the image was taken.

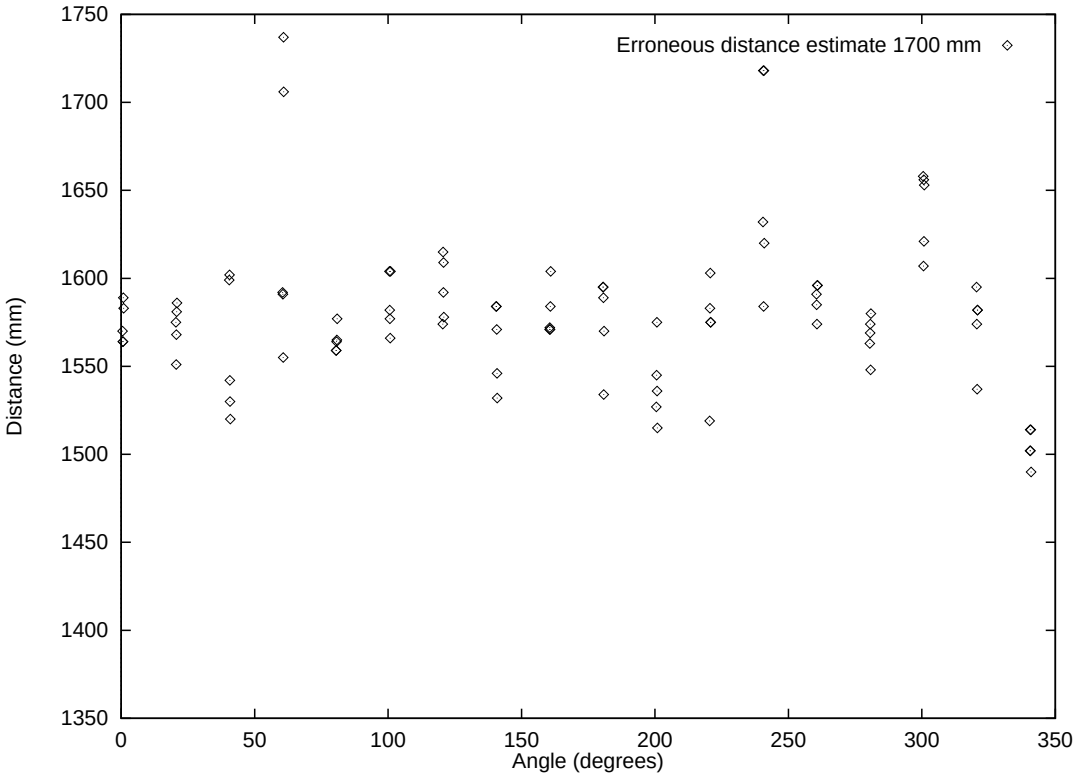
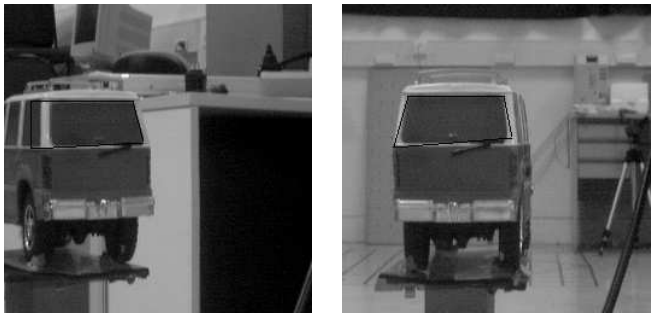


Figure 15: Scatter plot of the distance predicted by pose determination against the angle from which the image was taken when the prior estimate of pose was 1700mm (true value was 1550mm).

(a) (b) (c)



(d) (e)

Figure 16: Moving around the third corner of the model car object.

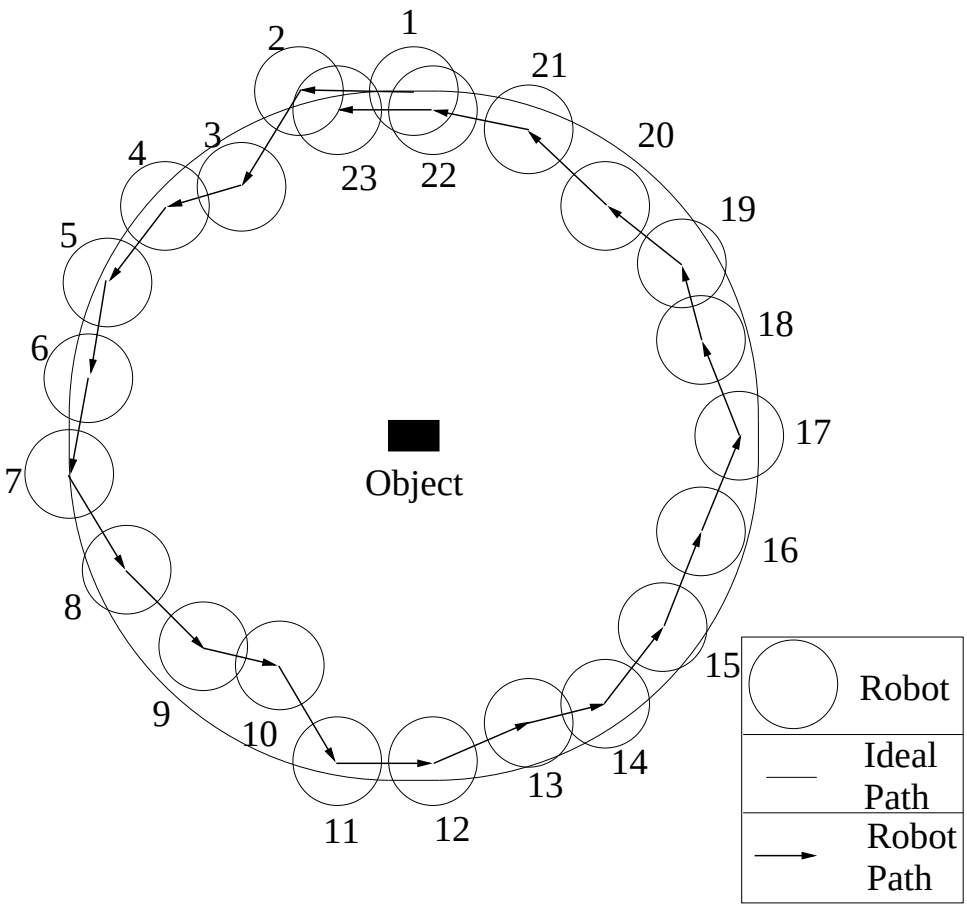


Figure 17: Circumnavigating the power supply. This diagram is to scale, measurements are to the nearest 100 mm.

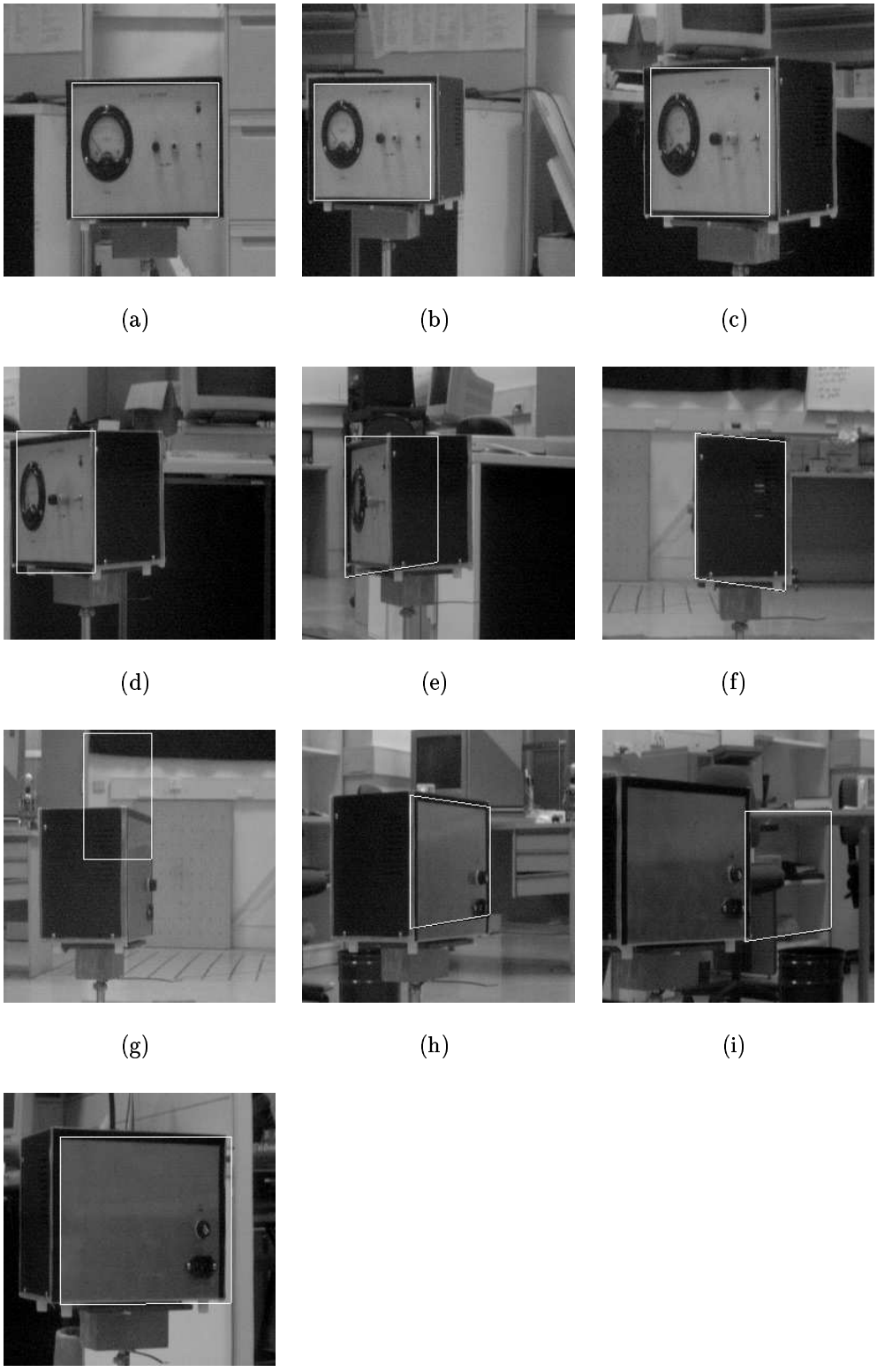


Figure 18: Circumnavigating the power supply.

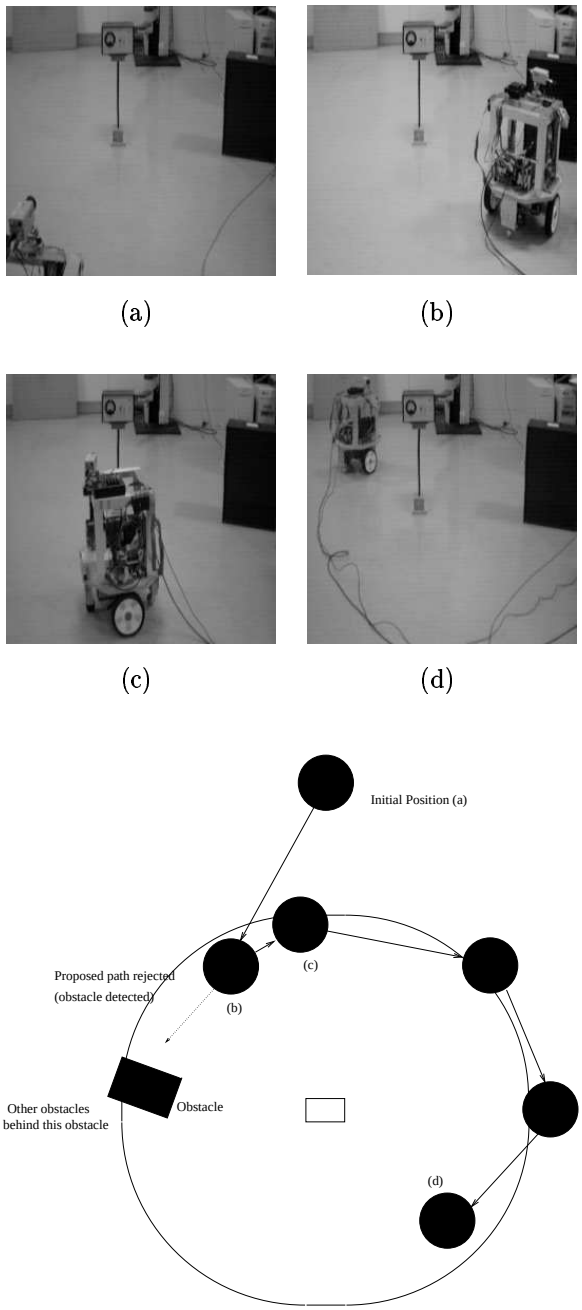


Figure 19: The robot's aim is to navigate around to the back surface of the power supply, while an obstacle blocks its path. (a) The starting position. (b) The robot detects that the obstacle (SGI workstation, the dark box in the image) is blocking its path for travelling counter-clockwise around the object. (c) The robot turns around and begins travelling clockwise around the object. (d) The robot at the final destination. (e) Schematic of the path taken.