Automatic Camera Placement for Robot Vision Tasks

Bill Triggs & Christian Laugier
LIFIA, INRIA Rhône-Alpes,
46 avenue Félix Viallet, 38031 Grenoble, France.
Bill.Triggs@imag.fr, Christian.Laugier@imag.fr

Abstract

Remote sensors such as CCD cameras can be used for a variety of robot sensing tasks, but given restrictions on camera location and imaging geometry, task constraints, and visual occlusion it can be difficult to find viewing positions from which the task can be completed. The complexity of these constraints suggests that automated, quantitative methods of sensor placement are likely to be useful, particularly when the workspace is cluttered and a mobile robot-mounted sensor is being used to increase the sensible region, circumvent occlusions, and so forth.

We describe a camera placement planner designed to produce heuristically good static viewing positions for a robot-mounted CCD camera in an experimental workcell. It can be configured to produce viewpoints for a variety of tasks such as workpiece location, inspection and modelling; feedback control by visual servoing; and task progress monitoring. The planner uses a novel probability-based global search technique to optimize a viewpoint evaluation function that heuristically combines task, camera, robot and environmental constraints. The main advances over previous work are the incorporation of kinematic accessibility and collision constraints in the viewpoint evaluation and the introduction of a search technique powerful enough to handle the resulting strong nonlinearities.

1 Introduction

Video cameras and other remote sensors are useful for a variety of robot tasks such as object location, inspection and modelling; visual servoing; and task progress monitoring. Correct sensor placement is indispensable for the successful execution of these tasks, so it makes sense to mount the sensor on a robot that can optimize the viewing position, extend the viewable region, circumvent occlusions, and so forth. However, the quality of a camera placement depends on a complex combination of task, camera, robot and environmental factors and the space of possible placements is quite large, so automated, quantitative methods of making sensor placement decisions are needed.

We have recently completed the first version of a planner designed to produce heuristically good static viewing positions for a fixed-focus robot-mounted CCD camera in our experimental two robot workcell. The planner is optimized for intervention-style robotics in potentially cluttered workspaces and can be configured for a variety of tasks. For example, it has been integrated into a visually-guided grasping system being developed under the European Esprit collaboration SECOND [4], where it is used to choose viewpoints for workpiece verification and pose correction and for visual servoing of the grasp approach.

Viewing positions are chosen by minimizing a heuristic viewpoint evaluation function over a space of possible camera placements, using a powerful new subjective probability based global search technique. The following factors are taken into account in the assessment of overall view quality:

- **Task:** viewing direction and distance, coverage.
- **Camera:** field of view, focus.
- **Robot:** kinematic reachability of camera pose.
- **Environment:** robot-environment collision, task occlusion and clutter.

There have been several previous studies of automated camera placement. Cowan [6, 7] developed geometric models of occlusion and camera resolution, focus and aperture constraints, and also considered simultaneous camera and light source placement, as did Yi [13]. Tarabanis and Tsai [10, 11] also took a constraint based approach and developed a sophisticated model of workpiece self-occlusion for their MVP system. Cameron and Durrant-Whyte [5] discussed more general camera placement issues from a Bayesian decision-theoretic point of view. Our present approach is oriented rather more towards robot planning and can be viewed as a sequel to the work of Al-Chami [1, 2].

Our current task and camera models are not as sophisticated as those of Cowan and Tarabanis & Tsai, although they are gradually being refined. The main contributions of the present work are the incorporation of kinematic accessibility and collision constraints into the placement evaluation and the introduction of a new global search technique capable of handling the intricate constraint geometry that results. This bias was largely determined by our interest in cluttered and uncertain workspaces in the context of intervention robotics, where collisions are likely and 'classical' geometric task and constraint modelling seem less appropriate. It must be emphasized that the inclusion of kinematic and workspace constraints makes the placement problem significantly harder: the optimization of even a trivial function over a region as complex as a robot configuration space is a difficult task, and a viewpoint quality metric is by no means trivial.

2 Planner Architecture

Our planner is designed to produce a heuristically good static **viewing pose** (*i.e.* 3D viewing position and orientation) for a calibrated robot-mounted camera observing a predefined task

[0]To appear in *IEEE International Conference on Robotics & Automation*, Nagoya, Japan, 1995.

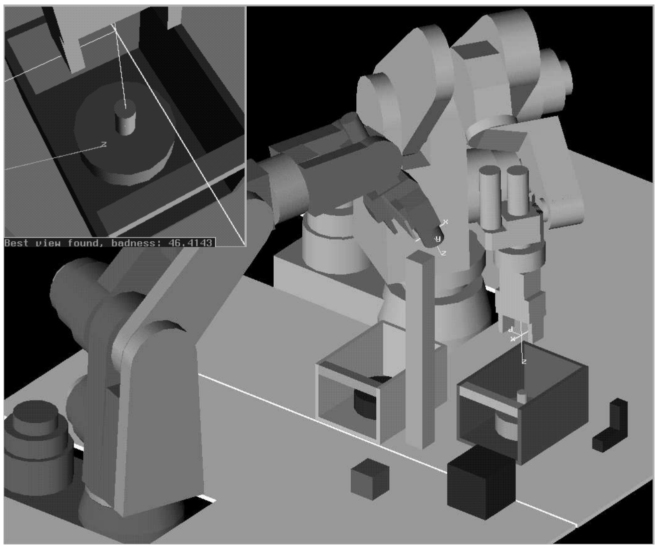

Figure 1: An optimal camera placement found by our planner, and (inset) the corresponding predicted image.

in a known workspace. The pose is to be reachable and collision free and must provide the camera with a clear, well-proportioned, in-focus view of the scene features relevant to the task, without occlusion or too much peripheral clutter. This typically involves trade-offs — perforce heuristic — between several conflicting classes of constraint, so it makes sense to package the complexity and the associated weighting parameters in a **view pose evaluation function** that returns a single heuristic measure of ‘overall viewpoint quality’ or ‘badness’.

The ‘best’ pose can then be found by classical search (function minimization) over some suitable space of candidate poses. In fact *any* ‘sufficiently good’ view should allow the task to be completed. Our simple static evaluator provides only a rough approximation to the rather difficult notion of ‘view quality for task completion’, so we do not insist on finding the absolute global minimum: any reasonable solution will do. Nevertheless, we will formulate the planning problem as a **heuristic function minimization**.

For simplicity we treat the evaluator as a ‘black box’ that can give point evaluations but no other hints as to likely places to search, and we restrict ourselves to a search paradigm that generates a stream of candidate view poses and evaluates each independently in turn, using the results to guide the generation process. The output is the best candidate found. This loosely-coupled scheme turns out to be fairly effective in practice despite its limited information flow. The decoupling greatly simplifies the driver and evaluator code and allows either to be replaced at will. In fact, our system is designed to provide a choice of search drivers, all using the same evaluator. Generate-and-test schemes like this also have a natural ‘anytime’ character in that the expected solution quality increases smoothly with search effort. Since we are prepared to accept any sufficiently good viewpoint we can simply stop the search as soon as a good enough candidate is found.

3 Optimization Techniques

Several features of the viewpoint evaluation function combine to make its optimization difficult:

- Evaluation is relatively expensive owing to the large amount of geometric computation required for image prediction, kinematics, and especially collision detection. The optimization technique needs to make the most of each function evaluation, even if this involves a substantial amount of subsidiary computation.
- The evaluation function is highly nonlinear and gives a rather convoluted ‘landscape’ with steep ‘cliffs’ where collision and kinematic accessibility constraints switch on, wide forbidden regions, and relatively shallow local minima lying in deep ‘valleys’ of accessibility. There are often several distinct valleys as occlusion and accessibility constraints can cut the scene in two. Essentially local techniques such as gradient descent, the simplex method and simulated annealing are not well suited to such landscapes. They are difficult to initialize owing to the wide forbidden regions, and apt to fall into poor local minima and then waste a lot of time locating them very precisely. Moreover, the range of gradient scales and the general twistedness tend to destabilize and slow the convergence of any technique that makes strong assumptions about function smoothness.
- A search over (an open subset of) all 3D camera poses would be six dimensional. This is sufficiently high to preclude simple ground covering methods such as grid search. If a global technique is to be used it needs to be well focused, and the search dimension probably needs to be reduced. (In fact, we currently restrict the search to a 3D space of upright camera placements directed towards the task centre).

In view of the above it seems most appropriate to use a global search technique that maintains a *set* of active search regions and progressively refines them, using relatively sophisticated search focusing heuristics with some built-in knowledge of the type of functional landscape produced by the viewpoint evaluator. To meet these requirements we have developed a new approach to function optimization that uses subjective-probabilistic function variation models to focus search effort on the regions most likely to yield better function values. To speed the search, known properties of the evaluation function can be encoded directly in the probability distributions. For comparison we are also intending to implement several conventional local techniques using the same evaluation function.

4 Viewpoint Evaluation

This section describes our view pose evaluation routine. Its job is to examine a hypothetical camera placement from the point of view of task, camera, robot, and environmental constraints and to form an overall consensus on its potential quality as a viewing position. The evaluator must be adaptable to a wide range of tasks and situations, and its internal parameters — particularly those governing the heuristic trade-offs between different aspects of view quality — must be intuitively meaningful and easily adjusted.

For simplicity, evaluation is structured as a series of independent heuristic tests. Each test examines a single ‘atomic’ aspect of view quality and outputs a nonnegative 'badness' score. Values near zero indicate an ideal situation and those greater than one a marginal one. The results are weighted to reflect their test's relative heuristic importance and summed to produce an overall pose badness metric as if they were independent χ^2 variables. Binary tests such as kinematic accessibility and collision detection return either zero or an essentially infinite value, but optional heuristic terms can be switched on to guide the search into feasible regions and produce a controlled aversion to near-infeasible situations.

For speed the tests are arranged roughly in order of computational cost and candidates whose running badness score becomes unacceptably large are discarded immediately. This is particularly important in the initial stages of the search, when many inaccessible and otherwise bad poses are tried before the more promising regions of the search space are discovered.

Currently the viewpoint evaluator performs the following tests:

- Robot kinematics comes first as many other tests depend on it and can not be run if it fails. The evaluator supports search in task, end-effector or joint space and provides appropriate forward and inverse kinematics to fill in the missing joint and pose information. Note that our spherically-wristed SCEMI robots have relatively inexpensive closed-form kinematics.

- The camera position is checked against a list of rough workspace bounds (clipping planes). This quickly eliminates placements with the camera underneath the work table or behind a wall, without the cost of full interference detection.

- Simple heuristics encoding camera focal and resolution constraints are used to evaluate the task-camera distance. (A more rigorous focus and resolution model would be desirable [6, 7]). The angular offset of the camera axis from the task origin is also determined, to help pull the line of sight towards the task.

- Task-dependent constraints on the viewing position are evaluated. The task is characterized by a polyhedral region that must be visible, a local frame defining a task origin and coordinate system, and a dedicated task viewing position evaluator that encodes the particular task's intrinsic viewing angle and distance requirements, irrespective of the camera and the surrounding environment. (Currently, this a simple angle-and-distance heuristic, but it could be extended to include task self-occlusion as in [10, 11]).

- The imaging geometry is modelled by a camera calibration (projection matrix) and a corresponding set of image clipping planes. The potential visibility of the 3D task region is evaluated by predicting its image, *i.e.* by projecting its silhouette outline into the image plane using the known camera pose and calibration, and then clipping against the known image borders. Invisibility is penalized according to the percentage of the projected task area that is clipped. For ease of implementation we currently project only a polyhedral approximation to the convex hull of the task region.

- The environment is searched for occluding and cluttering objects. Objects lying between the camera and the task region physically obstruct the view of the task and contribute to occlusion. Objects lying behind or near to the task region in the image contribute to clutter. Although clutter does not actually obstruct the view of the task, it makes visual routines such as segmentation difficult because the task background is too 'busy' and confusing, so it also needs to be penalized. To evaluate clutter and occlusion, polyhedral object models are projected into the image and compared with the projected task region. All objects lying near the task region contribute to clutter, and when the object and task images overlap an additional clutter or occlusion penalty proportional to the fraction of the task silhouette overlapped is charged, depending on whether the object lies behind or in front of the task region.

- Robot-environment interference detection comes last owing to its high cost. Our current system does not allow us to evaluate the degree of penetration so we simply return a yes/no answer.

5 Probabilistic Recursive Search

Now we give an outline of the function optimization technique used to minimize our viewpoint evaluation function. More details can be found in [12]. Roughly speaking, a region-refining global search process is guided by a probabilistic function interpolation heuristic. The intuitive idea is to focus search effort on regions that seem likely to yield a significant improvement over the best currently known value, given their size and measured function values. A sketch of the approach is as follows (see fig. 2):

- The search space is divided into a set of local regions defined by the points at which the function has been evaluated. For example the regions could be delimited by a triangulation of the points or there might be a 2^d-tree of cuboidal regions with a function sample at each cube vertex.

- The search starts from a given set of initial regions and samples and proceeds by successively selecting a region, subdividing it and replacing it with the resulting subregions, with local updates to the region structure and function evaluations at any new sample points introduced.

- Estimates of 'typical' function behaviour are combined with measured function values in each region to build a **probabilistic function interpolation** or subjective probability distribution for the function value at each point of the region, given the measurements.

- These distributions can be used to choose which region to refine and where to subdivide it, as follows. The goal is to optimize the function, so a sample only 'succeeds' if it improves on the **best currently known function value** f_{best}. If the probability density for the function value at some point is $p(f)\ df$, the **expected gain** or improvement to f_{best} from a sample placed at that point is

$$\langle\text{gain}\rangle \;=\; \int_{-\infty}^{f_{\text{best}}} (f_{\text{best}} - f)\ p(f)\ df$$

This defines an implicit ranking of all points and regions in the search space: the most promising sample point is the one with the greatest expected gain, and the most promising region is the one containing the most promising sample point. In practice the optimal sample locations are often hard to find but reasonable guesses can be made. Regions are ranked according to the expected gains at these approximate locations, and the

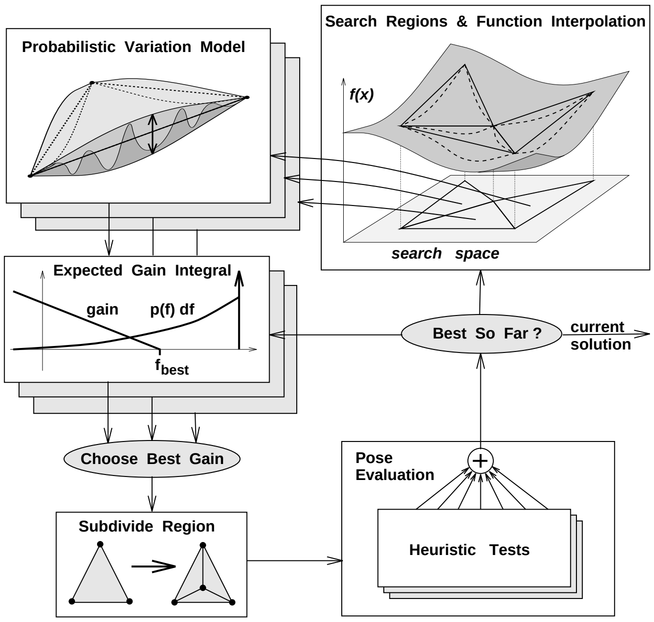

Figure 2: The basic probabilistic recursive search loop.

search proceeds by choosing the highest-ranked existing region at each step and subdividing it at its chosen sample location.

- Since the expected gain can only decrease as the best known function value improves, regions that currently look too unpromising can safely be deleted from the search.
- Subdivision continues until a sufficiently good sample has been found or a sufficiently thorough search has been made. Thresholding the minimum expected gain gives an effective termination predicate embodying a 'law of diminishing returns': search continues until significant further improvement seems unlikely.

The principal benefit of using subjective function value distributions is that they can very directly encode knowledge about likely function behaviour in a form immediately adapted to search control. *Any* relevant properties of the function and region can be encoded: upper and lower bounds, smoothness, Lipschitz (slope) bounds; size, shape, *etc*. The more restrictive and accurate the assumptions, the better focused the search will be.

The function value distribution at a point typically tails off fairly smoothly below some most probable value or 'mode'. This usually turns out to be greater than f_{best}, while the only part of the distribution relevant to the gain integral is that below f_{best}. In fact, the key feature of a value distribution is usually its asymptotic tail. The tails determine how quickly regions lying far above the current best function value are discounted from the search, so they have a strong influence on overall search complexity. By contrast, regions not far above the current minimum are likely to receive significant attention whatever the shape of their distributions. For this reason we will concentrate mainly on capturing the asymptotic form of the function value distributions.

The inner search loop involves finding and subdividing the most promising of the many existing regions. For efficiency it is essential to use an optimal priority queue implementation to rank the regions. Moreover, the expected gain from a sample or region depends on the reference level f_{best} with respect to which it is measured, so the relative rankings change as f_{best} improves. These changes are potentially problematic, however in practice it turns out that they can be tracked lazily with little additional overhead [12].

5.1 Region Structures

To flesh out the above outline we need to choose appropriate structures for region definition and refinement and decide how to model the function behaviour within them. For the current application we have chosen to use a triangulation-based decomposition with simplicial regions and a function sample at each vertex. Subdivision is by arbitrary vertex insertion followed by local retriangulation. We have also experimented with 2^d-tree based decompositions that place a function sample at each hypercube vertex.

Simplicial region models provide a very solid basis for models of function variation as there is a unique linear interpolant for the function across each simplex. They are also relatively economical on function evaluations as only one new sample needs to be introduced during each subdivision. Moreover, since they allow sample points to be placed arbitrarily, they offer the maximum possible scope for the effective use of function evaluations, and they can potentially be hybridized with conventional local techniques to produce global methods with good local performance. On the other hand, fixed subdivision techniques like the 2^d-tree are easier to implement, more robust, and often significantly faster for simple problems as they have less geometric overhead.

When using simplicial decompositions it is important to ensure that the simplices remain well-proportioned, as very flat or thin simplices are numerically troublesome and provide poor bounds on the function behaviour within them (and hence poor search focusing). In particular it is essential to break up the long initial edges of the triangulation. The simplest way to break edges is to explicitly place subdivision vertices on them. The simplices containing the edge must also be bisected, with new edges from each of their vertices to the subdivision point. This approach works moderately well, but it tends to produce a somewhat uneven distribution of samples containing many high-degree vertices with large 'fans' of edges.

To guarantee a more even sample distribution, we prefer to use a subdivision in which the regions form a **Delaunay triangulation** [9, 3]. Delaunay triangulations have a remarkable spectrum of optimality properties that ensure that their simplices are locally as regular and well-proportioned as possible, so they give very good control of the function variation across each simplex. Sample points can be placed arbitrarily, but each insertion must be followed by a quasi-local **conflict region retriangulation** process that maintains the local optimality property and gradually deletes long edges and misshapen simplices. The Delaunay-based search method gives good (and rather beautiful!) results, but it is probably only suitable for relatively expensive evaluation functions (like ours) as the retriangulation process tends to be fairly time consuming.

5.2 Subjective Models of Function Variation

We have based our estimate of the subjective probability of finding a given function value at a point on the deviation of the value from the unique linear interpolant of the function across the simplex containing the point. Subjectively, we expect large deviations to be less likely than small ones, but the extent of the difference must depend on the region geometry and the variability of the function. The potential deviation is zero at each vertex (where the function value is known exactly), but increases as the sample point moves away from the vertex, so that large regions have more total variation than small ones. The estimated strength of this region-size dependency is the key parameter in the model as it determines how search attention will be divided between large poorly investigated regions and small relatively good ones, *i.e.* how risk prone or averse the search algorithm will be.

In detail, our function value distribution model is based on a probabilistic variant of the Lipschitz variation bound $|f(\mathbf{x}) - f(\mathbf{y})| \leq K\,||\mathbf{x}-\mathbf{y}||^\nu$, where K and ν are positive constants and $||\cdot||$ is some positive metric on the search space. In mathematics this is usually required to be a strict bound saying that the function variation across a region is no more than a fixed constant times a power of the region diameter. Here, we use it as a subjective estimate of the *probable* amount of variation in a region of a given size. Specifically, we suppose that there is a scale length L, a function scale F and a power law constant ν such that the subjective probability $p(\Delta f \mid \Delta\mathbf{x})$ of a function variation Δf in a region of size $\Delta\mathbf{x}$ is a function only of the dimensionless variation measure $|\Delta f|/\mu(\Delta\mathbf{x})$, where the **variation scale** $\mu(\Delta\mathbf{x})$ is defined by

$$\mu(\Delta\mathbf{x}) \;\equiv\; F\cdot\left(\frac{||\Delta\mathbf{x}||}{L}\right)^\nu$$

The idea is that large variations are still possible but are considered subjectively less likely, and that on the whole smaller regions will have less overall variation. The units F and L are included simply to make things dimensionless: the key parameters are the exponent ν and the asymptotic form of the subjective variation distribution $p(\,|\Delta f|/\mu(\Delta\mathbf{x})\,)$.

The **volatility index** ν characterizes the function's 'roughness' and determines how the expected variation decreases as the subdivision grows finer. For example a function of (mostly) bounded slope might have $\nu \sim 1$ so that on average a region half the size has about half the variation, while a less well behaved function with many cusps might have a fractional ν. Since we are working with deviations from the function's local linear interpolant, a very smooth function dominated by its second order Taylor series might have $\nu \sim 2$ (*i.e.* $||\partial^2 f/\partial\mathbf{x}^2|| \sim \mathcal{O}(\text{constant})$). The value of ν and the asymptotic form of the variation distribution determine how quickly regions of a given size and level above the current function minimum will be discounted from consideration as the search proceeds.

In any model of this form, the variation scale starts at zero at each vertex and grows monotonically and isotropically as the sample point moves into the body of simplex. The maximum expected deviation always occurs when the sample point is as far as possible from the nearest vertex. A sample placed at this point provides the maximum possible control over the function variation on the resulting sub-simplices, and when the slope of the linear interpolant can be neglected — as it can in the asymptotic limits of large deviation or small simplices — it also has the greatest *expected gain* (estimated improvement in best known function value) of any point in the simplex. When the slope can not be ignored the optimal sampling location depends on f_{best} and is less easy to determine, so for simplicity we always try to place samples at the points farthest from the nearest vertices. Asymptotically this is a good policy as the asymptotic search complexity is governed by the small-simplex limit.

Any nonsingular simplex determines a unique sphere passing through all of its vertices called the **circumsphere**. The centre of this sphere or **circumcentre** is equidistant from each vertex and — when it lies in the simplex at all — it is the unique point of the simplex farthest from any vertex. If the circumcentre lies outside the simplex, the maximally distant point is still generically unique but seems to be significantly harder to calculate. It can lie on a facet of any dimension between 1 and $d-1$ and can be equally close to anything from 2 to d nearest vertices. It could be evaluated by brute force enumeration of the possible 'active' vertex sets with some subsidiary geometry for each, but for our application this seems unnecessarily complicated and we have preferred to use a simpler placement heuristic: the sample is placed at the circumcentre if that lies inside the simplex, and otherwise it is placed at the midpoint of the longest edge of the simplex. This ensures that elongated simplices are quickly broken up and is very effective at stabilizing the subdivision process. We have tried several other plausible heuristics, but in dimensions greater than 2 or 3 they rapidly lead to the production of singular simplices: one of the surprises of this work was how sensitive the high dimensional Delaunay retriangulation process is to non-randomly-placed insertions[1].

We also need to choose a suitable functional form for the deviation distribution $p(\,|\Delta f|/\mu(\Delta\mathbf{x})\,)$. In the absence of further information this choice must be made heuristically. This is unfortunate as the asymptotic form of the distribution has a significant influence on search efficiency. Typically some tractable smoothly decaying rule such as an exponential, Gaussian or power law is postulated. In the present case we decided that a roughly exponential form seemed as plausible as any other[2]. However, to ensure good search focusing the distribution must be well adapted to the actual function behaviour. In our application the view evaluation function is always positive, so there is an upper bound on the maximum possible gain or deviation. To encode positivity without departing too far from exponentiality, we chose to model the subjective function value distributions with the linear-exponential form

$$p(\Delta f \mid \Delta\mathbf{x}) \;\sim\; (f - f_{\text{min}})\cdot e^{-|\Delta f|/\mu(\Delta\mathbf{x})}$$

[1] The problems are particularly severe near the search region boundaries. It is essential to use a Delaunay update routine that is robust to insertion anywhere on the boundary.

[2] Recall that the asymptotic tail of the distribution is its most important feature. An exponential tail encodes the notion that each additional increment of variation is equally unlikely, while for a Gaussian tail the increments themselves become exponentially unlikely. Gaussians tend to produce a rather sharp gain cut off that might be deemed too model sensitive.

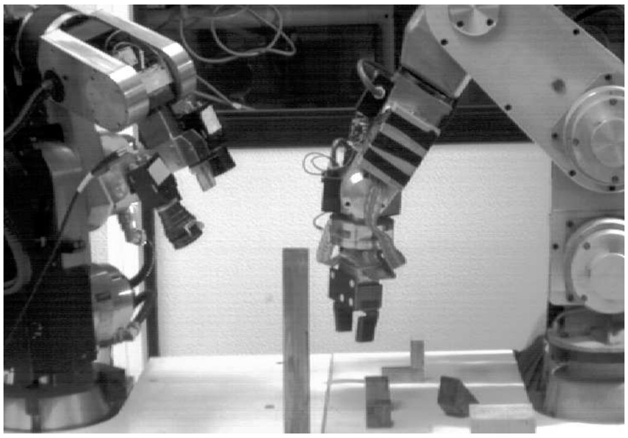

Figure 3: Visual grasp servoing from a planned viewpoint.

where $f \geq f_{\min}$ and $f_{\min} \geq 0$ is a strict lower bound on the function value in the region.

The above arguments are clearly rather subjective and many alternative function variation and sample placement models are possible. The key point is that the models chosen produce reasonably plausible and tractable estimates of the range of function values that might occur in a region, that can be used to concentrate search effort on relatively promising regions.

6 Implementation & Experiments

Our planner is written in C and runs under the ACT robot modelling system [8] as part of our integrated workcell demonstration [4]. The results of a typical run of the system are illustrated in fig. 1. The task was simply to get a clear view of the central peg in the right hand box. The search was over a large 3D space of upright camera placements directed towards the peg and took about 10 seconds on a Silicon Graphics R4000 workstation. The problem has several deep local minima owing to various occlusions, and the final pose is near the limit of the robot's reach. The pose found for the camera-carrying (left hand) robot is shown in the main panel while the predicted image is shown in the inset. Fig. 3 shows a grasp approach being visually servoed from a viewpoint chosen by the planner.

7 Discussion & Future Work

The initial version of the planner seems to be reasonably effective and we are happy with the general paradigm, although there are many small improvements that could be made. The task and camera models are rather heuristic and need to be upgraded. The search could probably be speeded up significantly by an initial infeasible region pruning process and better evaluation function bounding heuristics. In fact it is not really clear that the evaluation function is expensive enough to justify the full power and weight of the Delaunay-based search method: the 2^d-tree based technique may be faster in practice. Finally, the current *pose* planner needs to be integrated with a *path* planner as there is not much point in planning viewpoints you can not move to. (The necessary routines already exist in our system but they are not yet interconnected).

8 Summary

We have described an automatic sensor placement planner for a robot-mounted CCD camera viewing a robot task. The planner minimizes a heuristic viewpoint evaluation function combining task, camera, accessibility and occlusion constraints, over a space of possible camera poses. It is based around a global region-based search technique guided by a novel probabilistic heuristic that builds subjective models of the probable range of function values within each region and uses them to concentrate search on regions likely to yield a significant improvement in the best known function value.

Acknowledgments

This work was supported by the European Community through Esprit programs HCM and SECOND.

References

[1] O. Al-Chami. *Contribution à l'intégration Robotic/Vision en Manipulation Automatisée: Modélisation de la Tâche, Placement d'une Caméra Mobile et Localisation Fine d'Objet*. PhD thesis, LIFIA, Grenoble, 1994.

[2] O. Al-Chami and C. Laugier. Stratégie perceptive pour positionner une caméra. In *Proc. AFCET Reconnaisance des Formes et Intélligence Artificiel*, volume 1, pages 617–22, 1994.

[3] F. Aurenhammer. Voronoi diagrams — a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, 1991.

[4] C. Bard, C. Bellier, J. Troccaz, C. Laugier, B. Triggs, and G. Vercelli. Achieving dextrous grasping by integrating planning and vision based sensing. *Int. J. Robotics Research*. Accepted, to appear in 1995.

[5] A. Cameron and H. Durrant-Whyte. Optimal sensor placement. *Int. J. Robotics Research*, 9(3), 1990.

[6] C. K. Cowan and A. Bergman. Determining the camera and light source location for a visual task. In *IEEE Int. Conf. Robotics & Automation*, pages 509–14, 1989.

[7] C. K. Cowan and P. D. Kovesi. Automatic sensor placement from vision task requirements. *IEEE Trans. Pattern Analysis & Machine Intelligence*, 10(3):407–16, May 1988.

[8] E. Mazer, J. Troccaz, and *et.al.* ACT: A robot programming environment. In *IEEE Int. Conf. Robotics & Automation*, pages 1427–32, Sacramento, California, April 1991.

[9] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, 1985.

[10] K. Tarabanis and R. Y. Tsai. Sensor planning for robotic vision. In O. Khatib, J. Craig, and T. Lozano-Pérez, editors, *Robotics Research 2*. MIT Press, 1992.

[11] K. Tarabanis, R. Y. Tsai, and P. K. Allen. Automated sensor planning for robotic vision tasks. In *IEEE Int. Conf. Robotics & Automation*, pages 76–82, 1991.

[12] B. Triggs. Global optimization using probabilistic models of function variation. Technical report, INRIA Rhône-Alpes, Grenoble, France, 1995. (To appear).

[13] S. Yi, R. M. Haralick, and L. G. Shapiro. Automatic sensor and light source positioning for machine vision. In *10th Int. Conf. Pattern Recognition*, pages 55–9, June 1990.