The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Issues and Solutions in Surveillance Camera Placement

Duc Fehr, Loren Fiore and Nikolaos Papanikolopoulos
{fehr, fiore, npapas}@cs.umn.edu
Department of Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55455

Abstract— **Cameras are becoming a common tool for automated vision purposes due to their low cost. Many surveillance and inspection systems include cameras as their sensor of choice. How useful these camera systems are is very dependent upon the positioning of the cameras. This is especially true if the cameras are to be used in automated systems as a beneficial camera placement will simplify image processing operations. Therefore, a reliable positioning algorithm can lower the processing requirements of the system. In this paper several considerations for improving camera placement are investigated with the goal of developing a general algorithm that can be applied to a variety of systems. This paper presents this algorithm for placement problem in the context of computer vision and robotics. Simulated results of our method are then shown and discussed, along with an outline of future work.**

I. Introduction

The ratio between the amount of information that can be collected by a camera and its cost is very high, which enables its use in almost every surveillance or inspection task. For instance, one may argue that we can place thousands of cameras since they are cheap. The downside however, is that monitoring them can be very tricky and cumbersome. In these cases, it is not feasible for a group of human operators to simultaneously monitor all the cameras effectively. Programs have to be developed to help the operators succeed in their tasks. These programs are faster and more accurate when the surveilling cameras are placed appropriately.

Cameras can be mounted on mobile autonomous robots for surveillance or scouting purposes. These robots will be inexpensive in comparison to other robots using more complex sensors such as lasers. A system that could control the placement of these mobile robots in order to collect the largest possible amount of information would enhance the robots' usefulness.

Cameras can also be used to increase efficiency in automated assembly lines where repetitive tasks may cause a human monitoring the line to become bored or fatigued and thus miss critical errors. Replacing humans with automated systems for these tasks will increase the production speed of the assembly line and reduce the risk of missing faulty products, but in order for such an automated system to have good results, the system has to have the best possible view of the products it is monitoring.

The effectiveness of these camera systems is heavily dependent upon their physical placement. Thus, it seems advantageous to dedicate computational effort to determine the optimal viewpoints for these systems. However, camera placement is very dependent on the task the camera system has to perform and therefore any placement system that ignores the task at hand will not do well. For example gait classification needs different visual cues than face recognition. The goal of this work is to assess different cues that are important for different tasks and give mathematical functions to quantify these features. It is hoped that the simple and parametric functions found for these few example situations can be used in other situations since the features are thought to be universal to the placement problem.

After the discussion of related work, this paper will discuss the quality functions that were developed to describe the placement problem for three different tasks. Results from simulated placements using this algorithm will then be presented and areas of future research will be discussed.

II. Related Work

As early as 1987 O'Rourke described the placement problem in [1], where he discusses how to place guards in order to cover all the edges of a polygon. This constitutes the so called *Art-Gallery* problem. The proposed algorithms give the necessary positions of guards in order to achieve this task. However, the assumptions are strong, such as the assumption of a 360 degree field of view for the cameras as well as no image degradation with distance.

In [2], Cowan and Kovesi describe a method of computing the optimal camera placement for different task requirements. Cowan and Kovesi show that geometric relationships can be found between these task requirements and the camera locations. In this approach each requirement is considered individually first, and the 3D region of viewpoints that satisfy this requirement is calculated. The intersection of the different regions gives the set of acceptable camera locations.

Similar to Cowan and Kowesi in [2], Tarabanis *et al.* [3], [4] develop strategies that achieve an optimal placement with a certain degree of satisfaction. This placement strategy is based on four different constraints that are translated into equations. These are used to build a cost function that expresses the degree of satisfaction of a placement. Abrams *et al.* [5] extends this work from the static to the dynamic case by recomputing the static constraints at each time step.

A more recent placement strategy was devised by Chen *et al.* [6], [7]. Chen introduces a metric that allows for the measurement of the 3D position uncertainty of a moving

target. The quality metric that has been devised includes a probabilistic occlusion model, which constitutes the main contribution of Chen's work.

Another strategy that also uses a probabilistic modeling of occlusions was developed by Mittal and Davis [8]. In this work probabilistic models are used in order to analyze the average cases whereas a deterministic approach is used in order to analyze worst case scenarios.

All these strategies assume *a priori* knowledge of the environment. Cowan and Tarabanis also assume some previous knowledge about the observed object itself. The strategy of this paper is different from these approaches in that it has a minimal set of *a priori* knowledge about the target and no *a priori* knowledge of the environment. It gains its knowledge from observation and uses this knowledge in order to place the cameras. Three different tasks will be discussed in the following section and general shared features will be found that can be used for defining the placement problem.

III. Camera Placement

A. Description

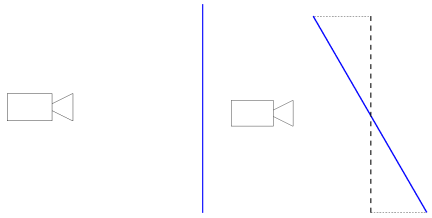
We investigate the possibility of introducing a quality metric for camera placement that can be used for three different tasks: gait classification, face recognition, and people counting.

Gait classification is the task of telling apart gaits such as walking and running given a series of test images from a target. In [9] and [10] gait classification was investigated. In these papers, the cameras are to be placed to facilitate the algorithm to classify gaits. This means being able to see a person from the side at a shallow angle to be able to observe the stride. Thus, in these papers, the cameras were placed perpendicularly to the walking direction.

Face recognition is the task of matching a face in an image to a database of sample face images. In [11] Brunelli and Poggio investigate face recognition. Similar work has been done by Pentland *et al.* in [12]. In these two cases the people have to be photographed from the front in order to be able to make comparisons. The more a camera moves to the side, the less reliable the algorithm becomes. Face recognition algorithms try to be robust to these angle changes, but their task is greatly facilitated if the camera was placed well before the actual processing.

People counting is the task of estimating the number of people in crowded groups. In [13], [14] and [15] work has been done in this area. In all of these articles, the best view for cameras is directly above the crowd of people. This point of view tries to eliminate one of the main problems in group counting, which is self-occlusions of the people within the group.

To summarize, gait classification needs placement perpendicular to the gait direction and face recognition requires a placement facing the gait direction. Both of these need a shallow angle at the target in order to get relevant information from the images. People counting on the other hand works best if the camera is placed above the target.



(a) Without foreshortening (b) With foreshortening

Fig. 1. This figure the foreshortening effect. When the camera is placed perpendicularly at the desired object in blue (1(a)) it is able to collect better information than when the camera points obliquely at the object (1(b)).

As can be seen, these tasks require three different placement strategies. However, there are enough similarities between them that a single framework can be developed that solves all three. For example, both face and gait recognition require a good resolution image of the subject from the correct angle. The angle is thus a variable of the placement.

The setup for the placement system is as follows. First, we install a main calibrated camera that observes the entirety of a scene. From this camera's images we extract the movement of people through the scene by using motion tracking. Specifically we use the method from [16] but any other form of tracking people would work. Once we have this information, we use quality functions to determine the placement of mobile cameras that enable the realization of the task.

B. Quality Function

We are investigating three different effects: foreshortening, ground coverage, and resolution.

1) Foreshortening: The foreshortening effect impacts the quantity of information that can be extracted from an observation. Fig. 1 gives a schematic view of this point.

We observe that a projection into the perpendicular plane to the normal of the camera gives the measure of information we can gather. This is why we are using a cosine in Eq. (2).

The way the algorithm manages the foreshortening is the following. Once the trajectories of the people in the scene have been extracted, the data is discretized. We are assuming that people are moving on the ground plane and use a view from above projection onto the ground, so that we have 2D data to work with. To each point the direction of the moving person at that point is attached. Then the datapoints on the direction and on the position are clustered. Finally, we define vectors $\overrightarrow{a}$ and $\overrightarrow{N_a}$ as the mean vector of the cluster and the normal vector to the mean vector, respectively (Fig. 2). The vectors $\overrightarrow{a}$, $\overrightarrow{N_a}$ and $\overrightarrow{a} \times \overrightarrow{N_a}$ form a basis in 3D space.

Depending on which task is to be accomplished, a vector $\overrightarrow{v}$ is defined in this basis. This vector supports a cone that limits the visibility of the object to the camera. The angles γ and δ define the half angle of the cone that provides a good view for the camera and the half angle of the cone from the camera, respectively. The angles ϕ and θ give both the angle from the normal of both cones to the vector connecting the

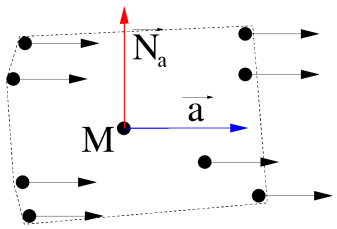


Fig. 2. This figure shows the vectors $\mathbf{a}$ in blue and $\mathbf{N_a}$ in red defined in III-B.1. The black dots correspond to the discretized position and the black arrows are the directions of the moving people. M is the mean point. The dashed lines correspond to the convex hull of the points in the cluster.

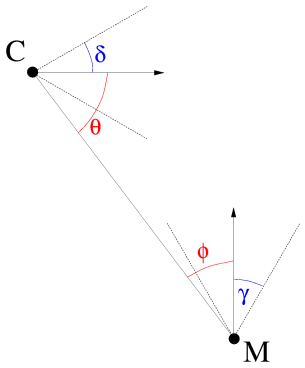


Fig. 3. This figure shows in a top view, the angles γ and δ, which are the half angles of the cones defined along with the angles ϕ and θ. Point C indicates the camera position and M the center of the observed cluster.

position of the camera and the mean cluster position. Fig. 3 gives a view on these angles.

For face recognition the cone supported by vector $\overrightarrow{v_f}$ is to be aligned with the direction of the path and thus

$$\overrightarrow{v_f} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^T.$$

For the people counting task the camera should be placed overhead as much as possible and thus the cone should be pointed upwards so that

$$\overrightarrow{v_p} = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T.$$

For gait classification the cone supported by vector $\overrightarrow{v_{g1}}$ should point perpendicularly to the moving direction and thus

$$\overrightarrow{v_{g1}} = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^T.$$

Since we can choose either side of the paths to place the cameras, we could equally use the opposite of this vector. This comes from the fact that for gait classification, it does not really matter from which side the target is observed. The classification algorithm will perform equally well on both sides and thus

$$\overrightarrow{v_{g2}} = \begin{pmatrix} 0 & -1 & 0 \end{pmatrix}^T.$$

Fig. 4 shows these different vectors in relation with the direction vector $\overrightarrow{a}$.

Once we have decided which cone to use, we can define the following function:

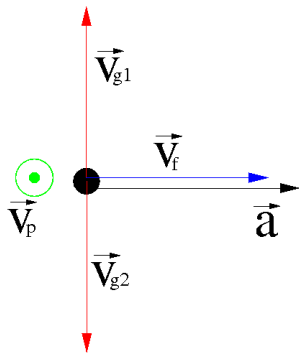


Fig. 4. This figure shows the vector $\mathbf{a}$ with the different vectors that define the cone orientation. Vector $\mathbf{v_f}$ in blue for face recognition, $\mathbf{v_p}$ for people counting in green and vectors $\mathbf{v_{g1}}$ and $\mathbf{v_{g2}}$ for gait classsification in red.

$$Q_F = \underbrace{g\left(\theta,\gamma\right)}_{\text{camera}} \cdot \underbrace{g\left(\phi,\delta\right)}_{\text{cluster}}, \tag{1}$$

where g takes arbitrary real numbers and is defined as,

$$g\left(x,y\right) = \begin{cases} \cos x & \text{if } |x| < y \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The function Q_F gives the quality function corresponding to the foreshortening.

2) Ground Coverage: Once we have computed the direction clusters and the foreshortening, we use this information again in the computation of the ground coverage term. In order to estimate the people density of a given region we take Parzen Windows [17] over the discretized data from the motion clustering. After this computation we build a convex hull around the thresholded data. These convex hulls define the areas of most human density in the scene. We are trying to maximize the coverage of these convex hulls with the camera frustum (Fig. 5(a)). In order to achieve this, the frustum is projected onto the ground plane. From there the differences and intersections between the convex hulls and the projected frustum can be computed (Fig. 6).

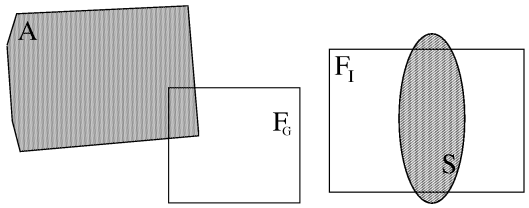
Placing the camera closer to the convex hull can achieve this. However, getting too close to the target entails that the frustum might not encompass the entire convex hull, which would mean that even though good information would be available locally, some parts of the hull would not be seen at all.

The ground coverage function takes into account these observations.

$$Q_G = \exp\left(-\beta_G \left(\frac{\max(F_G - A, A - F_G)}{F_G}\right)^{\alpha_G}\right) \tag{3}$$

where A is the area of the computed convex hull as shown in Fig. 2 (dashed lines) and F_G is the area of the frustum of the camera projected onto the ground plane.

The function behaves like a normal function with the most weight when $F_G - A = 0$. In the case of the whole convex hull being covered by the frustum, the maximum of the



(a) Projection into world ground plane (b) Projection into image plane

Fig. 5. This figure shows the different projections that are done in order to compute the quality functions. 5(a) shows the case for the ground coverage term where the frustum is projected onto the world ground plane (III-B.2). 5(b) shows the case where the object is projected into the image plane (III-B.3).

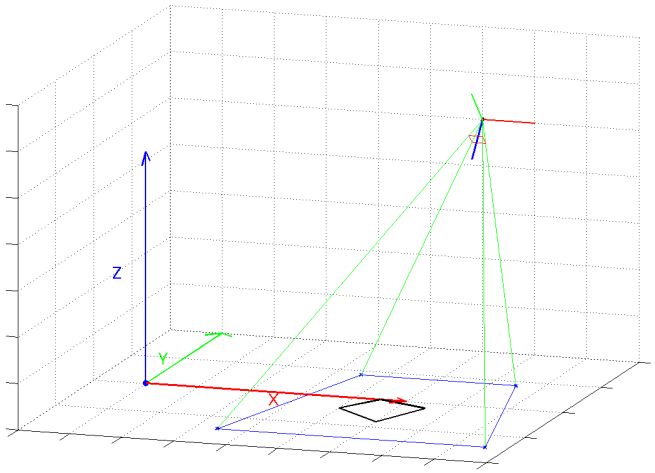
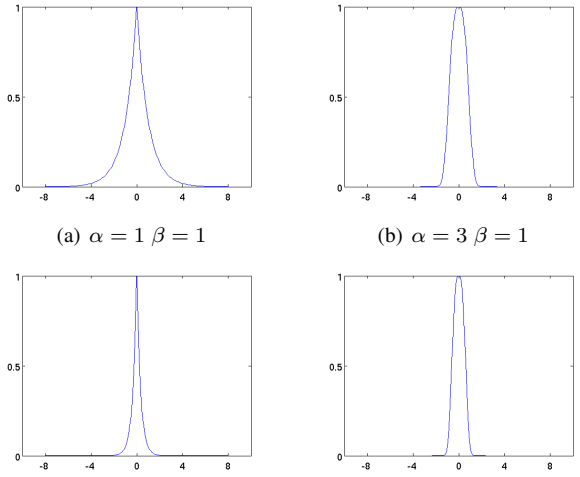


Fig. 6. This figure shows how the ground coverage gets analyzed. The world and camera frames are given in red (x-axis), green (y-axis) and blue (z-axis). The projected frustum in the ground plane is depicted in blue. The convex hull on the ground is represented in black.



(a) $\alpha = 1\ \beta = 1$ (b) $\alpha = 3\ \beta = 1$

(c) $\alpha = 1\ \beta = 3$ (d) $\alpha = 3\ \beta = 3$

Fig. 7. This figure shows the quality function Q_G for two different values of α_G and two different values of β_G.

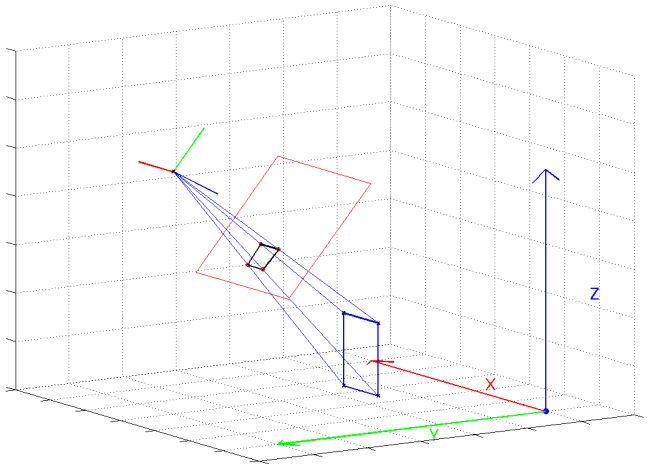


Fig. 8. This figure shows how the resolution gets analyzed. The world and camera frames are given in red (x-axis), green (y-axis) and blue (z-axis). A model (blue) is projected into the image plane (black). From these projections, the computations of the areas are done.

function is reached. If the convex hull completely encompasses the frustum, or the frustum completely encompasses the convex hull without filling the entire frustum, the function gives a smaller result. The most desirable would be for A to completely fill F_G.

The value α_G changes the shape of the function depending on the steepness the cost function is desired to have and the value β_G controls its width (Fig. 7).

3) Resolution: In order to define the resolution quality function, we first build a simple model of a person. We assume each person is a cylinder of height h and radius r that we place on each of the discretized points from section III-B.1. The values of h and r come from the known average sizes of a person, and are the only *a prior* knowledge required by the proposed algorithm. These cylinders are projected into the image plane of the camera and from there, a similar idea as in Section III-B.2 is used (Fig. 8). We try to cover the complete field of view of the camera with the projection of the cylinders. If the camera is too far away, and the surface of the target is smaller than a threshold t, we set the function to zero.

The function we use is the following:

$$Q_R = \begin{cases} \exp\left(-\beta_R \left(\frac{\max(F_I - S, S - F_I)}{F_I}\right)^{\alpha_R}\right) & \text{if} \quad S > t \\ 0 \qquad \text{otherwise} \end{cases} . \tag{4}$$

where S is the surface of the projected cylinder onto the image plane and F_I is the projected frustum onto the image plane. α_R and β_R control the steepness and the width in a fashion similar to the function in III-B.2.

The closer S and F_I fit, the better information we get (Fig. 5(b)). It is in this way that we use a similar idea as in III-B.2.

The ground coverage and the resolution quality function both try to find a balance between being able to cover the entire data available and maximizing the gathered per pixel information. If the camera gets closer to the convex hull the per pixel information rises since getting closer

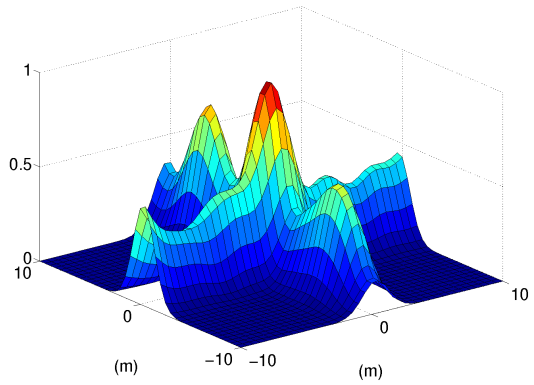
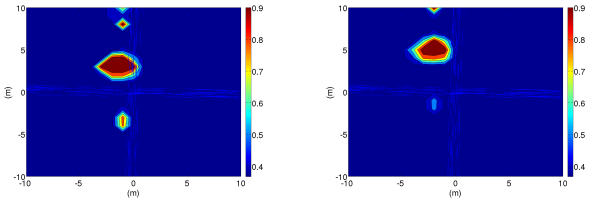
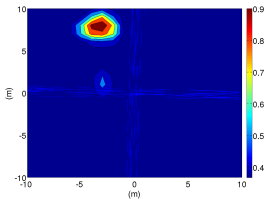


Fig. 9. This figure shows the human density estimation through Parzen windowing of the data. It is easily seen that the most dense region is the area where both main paths' directions cross.



(a) $h = 4$ (b) $h = 6$



(c) $h = 9$

Fig. 10. This figure shows the results of the experiment for ground coverage. In all of the figures the camera points towards the intersection of the path. The surfaces represent the quality of the positioning. The hotter the region, the better the positioning is. It can be observed that the different regions at the different heights line up with the direction of the camera.

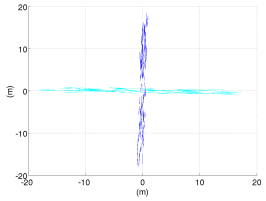
means that more details become visible. On the other hand however, even though the per pixel information becomes higher, getting too close to the target reduces the field of view of the camera and some information may be lost due to the fact that the camera cannot see the target in its entirety anymore.

Depending on the task that is to be accomplished we can put more emphasis on the ground coverage term or the resolution term by changing the different values of α and β.

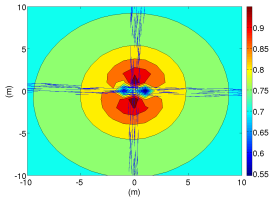
IV. Experimental Results

The experiments have been run on synthetic data in Matlab. The input data is superimposed on the different results in white (Fig. 10), blue (Fig. 11) and green (Fig. 12).

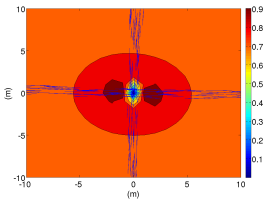
Fig. 9 shows the corresponding human density estimation. It can be seen that the area of most human density is where the paths cross. This is then in turn the area which the camera is trying to capture.



(a) Clustering result



(b) Gait classification positioning



(c) Face recognition positioning

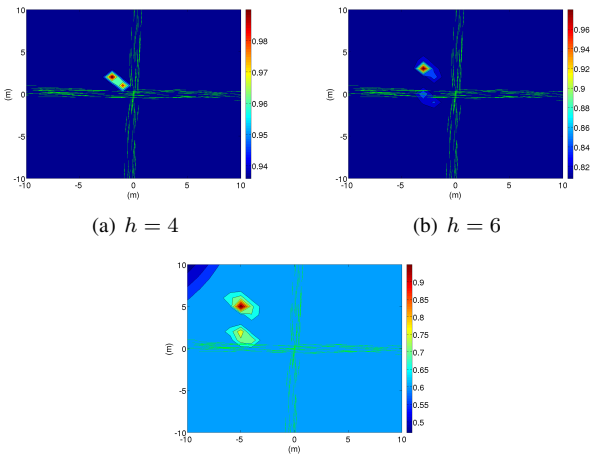
Fig. 11. This figure shows the results of the experiment for foreshortening. 11(a) shows the initial clustering step. 11(b) shows the positioning for gait classification and 11(c) shows the positioning for face recognition. It can be observed both placements are almost orthogonal one to another.

Fig. 10 shows the result of the ground coverage function. In all of the subfigures the camera points towards the main intersection of the path. The hot regions in these figures correspond to a good placement. It can be observed that the different regions at different heights line up with the direction of the camera, so that when superimposing them, the actual direction could be seen. The different smaller acceptable regions in Fig. 10(a) come from the shape of the human density regions. In Fig. 9 there are two other distinct peaks aside from the main peak which get captured in Fig. 10(a). In Figs. 10(b) and 10(c) these same acceptable regions appear, though they are less pronounced.

Figs. 11(b) and 11(c) show the result for foreshortening in the case of gait classification and face recognition, respectively. The same synthetic data was used as previously, on which clustering was performed to get the necessary angles for the computation (Fig. 11(a)).

Fig. 11 shows the impact of the foreshortening function. It can be observed that the good placements (hotter regions) for face recognition are orthogonal to the good placements for gait classification. A single camera will have trouble covering both clusters well at the same time however. Placing a camera in a good position defined by this quality function, it will be able to achieve its task well for one set of paths but less so for the second set of paths. This suggests that a more cluster centric approach might be appropriate, in which the use of a single camera per cluster becomes paramount.

Fig. 12 shows the resolution quality function for three different heights. The results are very similar to the ground coverage function. The camera points towards the intersection and through the different heights, a single ray can be followed. This particular angle is found because of the fact that at the intersection, more cylinders are placed and a camera scores a better result when its frustum is able to cover several projected cylinders at once.



(a) $h = 4$ (b) $h = 6$

(c) $h = 9$

Fig. 12. This figure shows the results of the experiment for resolution. The surfaces represent the quality of the positioning. The hotter the region, the better the positioning is. Similar to the ground coverage function, the camera points towards the intersection of the path. It is in this area where the frustum is most likely to be completely filled.

V. Conclusion

The aim of this work was the development of quality functions for camera placement that could be used for three different tasks. The main contribution of this paper is the idea of not only considering geometric constraints on the camera's position with respect to the discovered paths, but also take into account what the camera might see from a certain position. This is achieved in the resolution problem (III-B.3) by projecting simple models from the real world into the image plane and comparing with the coverage of the actual camera frustum in this plane. For the ground coverage problem (III-B.2) the frustum is this time projected onto the ground plane in the real world and this projection is then compared to the convex hulls of the different computed clusters. The results are promising in that they reflect the expected placement for the different tasks. It may become interesting to check how much information can be gained by using several cameras covering each cluster separately rather than having one camera trying to catch the entire scene by itself.

VI. Future Work

The next step is to quantize the amount of improvement in the different tasks gained from camera placement. This will constitute the next set of experiments. We will use some face recognition algorithms in different positions and check empirically that our placement improves the software's capabilities. We will do the same for gait classification and people counting algorithms. Another avenue of future work will be to attempt to use this formulation outside of the three applications for which it was designed. Great care was taken to make the equations as general as possible, and we believe this will allow the formulation to have expanded applications, but more work needs to be done to be sure of this. A third direction to explore will be the addition of probabilities into our models in order to get a description of the scene that may be more accurate by using better predictions of the movements and densities of the targets. Finally, as the main direction, the case of multiple cameras will be addressed in the context of cooperative sensor networks.

Acknowledgements

This material is based upon work supported in part by the U. S. Army Research Laboratory, the U. S. Army Research Office under contract number #911NF-08-1-0463 (Proposal 55111-CI) and the National Science Foundation through grants #IIS-0219863, #CNS-0224363, #CNS-0324864, #CNS-0420836, #IIP-0443945, #IIP-0726109, #CNS-0708344, and #CNS-0821474.

References

[1] J. O'Rourke, *Art Gallery Theorems and Algorithms*. New York: Oxford University Press, 1987.

[2] C. Cowan and P. Kovesi, "Automated sensor placement from vision task requirements," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 3, pp. 407–416, May 1988.

[3] K. A. Tarabanis, R. Y. Tsai, and P. Allen, "Automated sensor planning for robotic vision tasks," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Apr. 1991, pp. 76–82.

[4] K. A. Tarabanis and R. Y. Tsai, "Computing viewpoints that satisfy optical constraints," in *Proceedings of the IEEE Computer Science Society Conference on Computer Vision and Pattern Recognition*, June 1991, pp. 152–158.

[5] S. Abrams, P. K. Allen, and K. A. Tarabanis, "Dynamic sensor planning," in *Proceedings of the IEEE International Conference on Intelligent Autonomous Systems*, Pittsburgh, PA, Feb. 1993, pp. 206–215.

[6] X. Chen, "Design of many-camera tracking systems for scalability and efficient resource allocation," Ph.D. dissertation, Stanford University, 2002.

[7] X. Chen and J. Davis, "Camera placement considering occlusion for robust motion capture," Stanford University, Tech. Rep. CS-TR-2000-07, 2000.

[8] A. Mittal and L. Davis, "A general method for sensor planning in multi-sensor systems: Extension to random occlusion," *International Journal of Computer Vision*, 2007.

[9] L. Fiore, D. Fehr, R. Bodor, A. Drenner, G. Somasundaram, and N. Papanikolopoulos, "Multi-camera human activity monitoring," *Journal of Intelligent and Robotic Systems*, vol. 52, no. 1, pp. 5–43, May 2008.

[10] C. Bregler, "Learning and recognizing human dynamics in video sequences," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[11] R. Brunelli and T. Poggio, "Face recognition: Features versus templates," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Oct. 1993.

[12] A. Pentland, B. Moghaddam, and T. Starner, "View-based and modular eigenspaces for face recognition," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, Jun 1994, pp. 84–91.

[13] A. Schofield, P. Mehta, and T. Stonham, "A system for counting people in video images using neural networks to identify the background scene," *Pattern Recognition*, vol. 29, no. 8, pp. 1421–1428, 1996.

[14] K. Terada, D. Yoshida, S. Oe, and J. Yamaguchi, "A method of counting the passing people by using the stereo images," in *International Conference on Image Processing ICIP*, 1999.

[15] P. Kilambi, E. Ribnick, A. Joshi, O. Masoud, and N. Papanikolopoulos, "Estimating pedestrian counts in groups," *Computer Vision and Image Understanding, ELSEVIER*, vol. 110, pp. 43–59, Apr 2008.

[16] B. Maurin, O. Masoud, and N. Papanikolopoulos, "Monitoring crowded traffic scenes," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, Singapore, Sept. 2002.

[17] R. Duda, P. Hart, and D. Stork, *Pattern Classification*. Wiley Interscience, 2001.