Faster and More Accurate Face Detection on Mobile Robots Using Geometric Constraints

Michael Dixon, Frederick Heckel, Robert Pless, and William D. Smart
Department of Computer Science and Engineering, Washington University,
St. Louis, MO 63130, USA
{msd2, fwph, pless, wds}@cse.wustl.edu

***Abstract*— We develop a framework to allow generic object detection algorithms to exploit geometric information commonly available to robot vision systems. Robot systems take pictures with calibrated cameras from known positions and may simultaneously capture depth measurements in the scene. This allows known constraints on the 3D size and position of objects to be translated into constraints on potential locations and scales of objects in the image, eliminating potentially expensive image operations for geometrically infeasible object locations. We show this integration to be very natural in the context of face detection and find that the computational effort of the standard Viola Jones face detector (as implemented in OpenCV) can be reduced by 85 percent with three times fewer false positives.**

I. INTRODUCTION

Robot vision algorithms for object detection ought to be easy. Robots capture calibrated images from a known platform. They often operate in relatively simple environments and may capture partial or complete depth information as they image the scene. All these constraints should simplify the task of visually searching for a known object, but common robot vision systems do not exploit this geometric information.

We present a framework for adapting traditional object detection algorithms to make use of such constraints and demonstrate this on the problem of detecting faces. This is a key problem in Human Robot Interaction and one for which there are reasonable assumptions about the expected 3D position and size of the object. We transform these assumptions into geometric constraints that can be used to narrow the search space over which the face detector must operate. For example, we might reasonably assume that faces will only appear at human head height. Using knowledge of the location of the camera on the robot and fusing data from the robot's range sensors allows us to translate these world-space constraints into image-space constraints. These image-space constraints allow us to ignore areas of the image where no plausible face can appear, reducing the overall cost of face detection (see figure 1). Most existing face-detection algorithms, on the other hand, do not have this additional information available and must assume that a face can appear anywhere in the image, at any scale. This causes them to be more computationally expensive than necessary.

In this paper, we demonstrate that the geometric constraints representing our assumptions are easily combined with existing face-detection algorithms. We show that the constraints provide a significant reduction in the computational cost of face detection and an improvement in robustness to false positives[1].

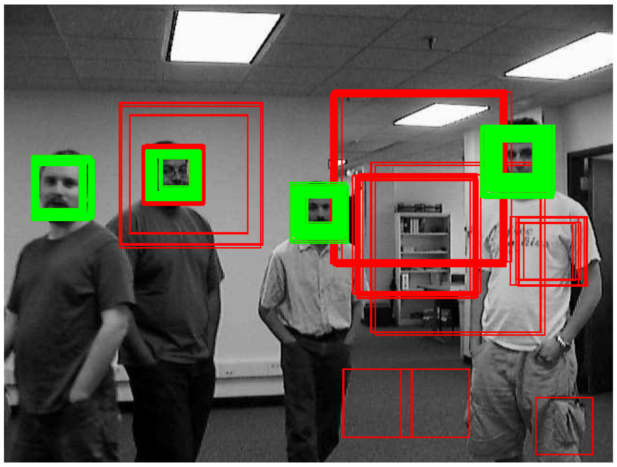

Fig. 1. Our method integrates geometric constraints available from robots with calibrated cameras and/or depth sensors with traditional face-detection techniques. In the above figure, all image regions identified as faces by the detector are highlighted. A calibrated laser range finder provides geometric cues to reject the red rectangles as inconsistent with the scale or position of natural faces. In fact, these image rectangles (and many others) never need to be tested at all. This leads to substantial performance benefits: a six-times reduction in processing and a three-times reduction in false positive rates.

We begin by briefly surveying existing work in on-line face detection, with a special focus on techniques that fuse data from additional sensors on a mobile robot platform. In section III, we give the derivation of the geometric constraints on the location and size of faces in image space. In section IV, we show the results of applying these constraints and demonstrate a significant improvement in performance and accuracy. Finally, we offer some conclusions and identify directions for future work.

II. PREVIOUS WORK

Current face-detection algorithms use a variety of techniques. The most common techniques make use of neural networks [1], color segmentation [2], local intensity distri-

[1]Note that if the assumptions are broken, this method may not discover some real faces; the particular contraints used should be tailored to individual situations.

butions and orientations [3], and machine learning techniques such as boosting [4], [5].

There has been some previous work in incorporating range data into face-detection algorithms on a mobile robot. Kim *et al.* [6] used stereo-based range and color segmentation cues to improve face detection. Byers *et al.* [7] used color segmentation and laser range data to find potential subjects in a robot photography system. Kleinehagenbrock [8] used laser range-finder data with a pre-processing step to find legs, combined with a skin segmentation algorithm. Blanco [9] used laser range-finder data to divide the image into subimages by first locating candidate leg locations. A neural-network based face detector was then used to locate faces in the subimages. Lang [10] made use of laser leg detection, face detection, and sound localization to direct the robot's attention.

Bellotto [11] used a combination of leg detection, tracking, and other techniques to increase the speed of face detection for a museum tour robot. Their technique used only leg or face detection to find people, rather than using the laser depth information to inform the face detector. This method could easily detect false positives when using the laser detector only.

While the use of leg detection is an improvement over using no sensor fusion at all, it does not generalize well to other depth sensors or to applications other than face detection. If a depth sensor is mounted in a location where legs are not visible, leg detection will provide no benefit. In addition, because we depend on depth information rather than shape, it is possible to apply the same principles presented here to the detection of other objects with known 3D properties, such as detecting cars or road signs in an autonomous vehicle.

III. APPROACH

In this section, we present our approach for integrating geometric constraints and explicit depth information into existing object-detection algorithms. In particular, we focus on face-detection on a mobile robot platform.

Most existing object-detection algorithms operate by exhaustively testing for the existence of an object at all possible positions and scales within an image. However, many objects in the real world have well-defined 3D properties that constrain the positions and scales at which they might appear in the image. For instance, human faces occur naturally in a small range of sizes and at a small range of distances above the ground plane. By using these geometric constraints to determine which subwindows might feasibly contain the object and discarding those subwindows which cannot, we can dramatically reduce the amount of computation performed by the actual face-detection algorithm as well as increase its robustness.

We begin by developing the geometric constraints, based on our knowledge of the geometry of the camera and our assumptions about the 3D positions of the faces to be detected. These constraints allow us to express each image region in terms of the range of depths at which a face might be found. We then show how to use measurements from two range sensors, a stereo camera and a laser range-finder, and what extra assumptions we must make to do this.

A. Geometric constraints

Each subwindow in the image corresponds to an infinite range of possible 3D positions and sizes. For example, because the depth is unknown, a particular subwindow might contain a small object close to the camera or a large object far away from the camera. However, with objects such as faces, there is a relatively small range of appropriate 3D sizes and positions. Most subwindows will only be consistent with these appropriate sizes and positions over a small range of depths, and some subwindows will not be consistent at any depth. In this section, we show how to compute the range of depths consistent with the known 3D properties of an object and how this can be combined with external depth measurement to determine which subwindows must be evaluated by the detector.

We begin by introducing our notation and coordinate conventions. We define our world-space coordinates to be aligned with the camera, such that the camera is located at the origin looking out along the z axis (with x and y being the horizontal and vertical axes, respectively). We define the position and orientation of the ground plane by its perpendicular unit vector, $\vec{n} = \langle n_x, n_y, n_z \rangle$, defining "up", and its distance from the origin, h. (Note that because the camera is located at the origin, h is simply the height of the camera above the ground.)

We represent the world-space position and size of a face as a square planar region, perpendicular to the z axis, centered at position (x, y, z) with a width of s. We likewise describe a subwindow of the image by its position, (u, v), and the width of the window, w, in pixels.

Given a subwindow, (u, v, w), we compute the range of possible sizes and positions as follows. We begin by applying the inverse of the camera calibration matrix,

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = K^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}, \tag{1}$$

where K is defined by the camera's focal lengths (f_x and f_y), center of projection (c_x, c_y), and skew (α) as follows:

$$K = \begin{pmatrix} f_x & \alpha f_x & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}.$$

This defines the following ray r along which the object must lie,

$$\vec{r} = \langle r_x, r_y, 1 \rangle = \langle x/z, y/z, 1 \rangle. \tag{2}$$

Similarly, based on the width of the subwindow, w, we compute the world-space size of the object relative to its depth,

$$r_s = s/z = w/f_x. \tag{3}$$

The constraint on possible object heights is defined by two planes parallel to the ground and positioned at the minimum and maximum feasible heights, h_{min} and h_{max}. We can determine the range of depths, D_h, consistent with these height constraints by solving for the intersection of the ray, r, with the two bounding planes as follows:

$$D_h = [-\frac{h_{min} - h}{\vec{n} \cdot \vec{r}}, -\frac{h_{max} - h}{\vec{n} \cdot \vec{r}}]. \tag{4}$$

Note that when $\vec{r}$ is perpendicular to $\vec{n}$, the denominator will be zero. In this case, we treat these values as either ∞ or $-\infty$ depending on the sign of the numerator.

The constraint on possible object sizes is defined by bounds on the minimum and maximum feasible sizes, s_{min} and s_{max}. We can solve for the range of depths, D_s, consistent with these size constraints as follows:

$$D_s = [s_{min}/r_s, s_{max}/r_s]. \tag{5}$$

The range of valid depths, D, consistent with both sets of constraints is given by

$$D = D_h \cap D_s. \tag{6}$$

We compute this range for each subwindow, (u, v, w), before evaluating it with the detector. If the intersection of these two ranges is empty, then there is no feasible 3D position and size associated with this subwindow at any depth. Such subwindows can be rejected without any evaluation by the detector. In section IV, we show that this allows the average computation performed by the detector to be cut in half.

Note that these ranges remain static as long as the height of the camera and its orientation relative to the ground plane (h and $\vec{n}$) do not change. Thus, these values only need to be recomputed when the camera is raised/lowered or is tilted or rolled. Furthermore, if we constrain the rotation of the camera so that it may pan and tilt but cannot roll, then we can disregard the values of r_x and N_x in equation 4. As a result, D_h depends only on v, just as D_s depends only on w. Thus we can save computation by precomputing the ranges of D_h for all values of v and D_s for all values of w and storing the results in a look-up table.

B. *Incorporating depth from external sensors*

In addition to having a known camera geometry, a mobile robot is usually equipped with sensors that measure the depth of objects in the scene. This additional depth information can be easily integrated with the geometric constraints outlined above. Our goal is to compute the range of depths, M, consistent with our sensor measurement at a given subwindow location, and to use this additional constraint to restrict the number of image regions that must be tested. We modify equation 6 to include this additional constraint:

$$D = D_h \cap D_s \cap M. \tag{7}$$

Note that when no external measurements are available, $M = [0, \infty]$. The following sections illustrate how to compute M for both laser and stereo depth sensors.

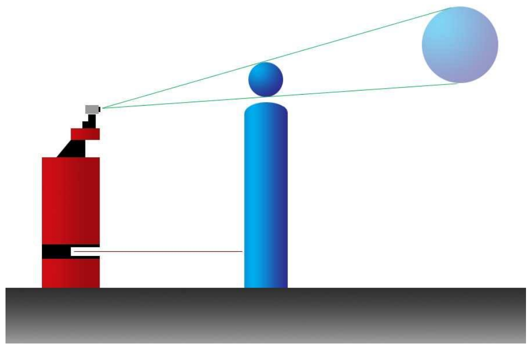

Fig. 2. A simple diagram of a mobile robot with a camera and laser range-finder. A particular image region corresponds to a number of possible object positions and sizes, depending on depth. Incorporating data from the laser eliminates this ambiguity.

Incorporating stereo data

A common method of measuring depth on a robotics platform is stereo vision. By applying a dense stereo matching algorithm to a pair of calibrated stereo images, it is possible to compute a measure of the depth at each point in the image. Such a measure, $depth_s(u, v)$, can be integrated into equation 7 as follows.

For each subwindow, (u, v, w), the set of depths consistent with the stereo depth measurement is

$$M = [depth_s\,(u, v) - \epsilon_s, depth_s\,(u, v) + \epsilon_s],$$

where ϵ_S is the tolerance of the stereo computation. This M is used in equation 7 to define the set of feasible depths.

Note that, in general, the disparity function is only defined over image regions with sufficient, unambiguous texture. Because faces satisfy these requirements, there is typically an accurate disparity measure for each face, especially around the edges. However, to prevent sparse data from introducing error, we fill small gaps in the disparity function by propagating nearby values. In practice, any image regions which remain unfilled after this operation lack sufficient texture to contain a face and can be safely ignored.

Incorporating laser data

In this section, we assume that the robot is equipped with a single eye-level camera and a laser range-finder similar to the popular SICK device that is nearly ubiquitous in research robot platforms. This device scans in a plane parallel to the ground plane and returns a number of equally spaced radial distance measurements (commonly 180 or 360) over the front 180 degrees of the robot. We assume that the range of camera pan values is restricted so that every possible image is covered by the laser range-finder.

Let the laser data be defined as a function $L(\alpha)$, which returns the distance to the nearest object along a particular heading, α. This function is defined over a discrete set of angles, A, corresponding to the individual distance measurements. Given the translation and rotation of the camera relative to the laser, we can compute the world-space coordinates at which each laser reading, a_i, intersects an object in the world, (x_i, y_i, z_i), and map each point to its corresponding location in image-space (u_i, v_i) using the calibration matrix, K. This allows us to define depth as a function of image coordinates for each laser reading,

$$depth_L(u_i, v_i) = z_i. \tag{8}$$

However, this function is only defined over the discrete set of values of u and v that correspond to laser range values. In order to generalize the distance measurements taken in the laser plane to the rest of the scene, we make a "footprint" assumption. That is, we assume all objects in the scene have a uniform cross-section through any plane parallel to the ground, and thus a depth measurement taken at the height of the laser can be generalized to any other height. This allows a single laser reading to define an entire column in $depth_L(u, v)$ rather than just a single point. The remaining undefined regions can then be filled in with the nearest defined value.

For each subwindow, (u, v, w), the set of depths consistent with the laser measurement is

$$M = [depth_L(u, v) - \epsilon_L, depth_L(u, v) + \epsilon_L],$$

where ϵ_L is the tolerance of the laser readings.

The footprint assumption: The footprint assumption is a common one in robotics, but it does not hold in all situations. In our application there are two types of failures that this assumption may cause. First, people do not, in general, have a uniform cross-section. At the height at which a typical laser range-finder is mounted (approximately 40 cm), the cross-section of a human is actually composed of two distinct parts (the legs). Thus, on occasion, the distance measurement associated with a given face may actually be the result of looking between the subject's legs to a more distant object. To correct for this, we perform a simple closing operation on the raw laser readings, which fills all gaps smaller than a given threshold with shortened readings.

Second, not all obstacles which obstruct the laser also obstruct the camera's view of a face. For instance, if a person is standing behind a table or desk, his or her face will be clearly visible but the laser will return the depth to the obstruction rather than to the person. This will cause the detector to erroneously ignore subwindows that may actually contain a face. In many HRI settings, this is actually an acceptable outcome; if the robot is only concerned with detecting people directly interacting with it in an open space, any persons standing behind obstacles can be safely ignored. However, in situations where such errors are unacceptable, we define, for each subwindow, (u, v, w), the set of depths consistent with the laser reading to be $M = [depth(u, v) - \epsilon_L, \infty]$. This will require the detector to test more image regions, particularly when $depth(u, v)$ is small, but will prevent laser occlusion from causing missed detections.

IV. RESULTS

To evaluate our approach, we combined it with the implementation of the popular Viola-Jones face-detector [4] available in the Intel OpenCV library [12]. The Viola-Jones face detector is a hierarchical detector that performs a series of increasingly expensive tests based on easy-to-compute, primitive features (Haar features). The tests are run in a cascade, with candidate regions being removed from consideration as soon as they fail a test. Regions that survive all of the tests are candidate faces, and dense collections of candidates are grouped and reported as a detected face. Since it is designed only to work in image space, with no additional sensor data, the algorithm must consider all possible positions and scales when searching for faces.

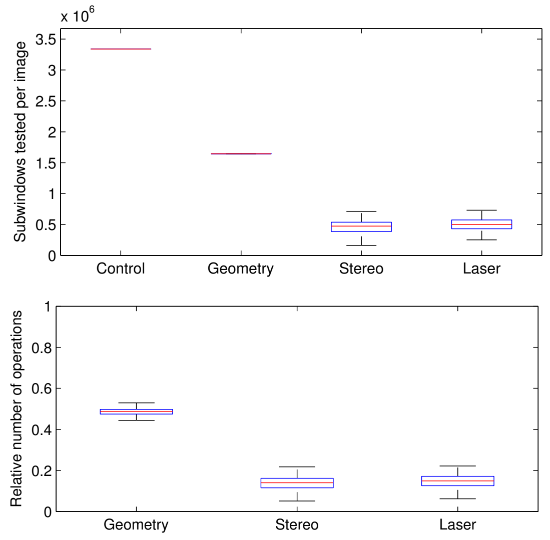

Fig. 3. By using geometric constraints to discard infeasible subwindows before they are evaluated by the detector, we are able to significantly reduce the amount of computation required. The upper plot shows the average number of subwindows that must be evaluated. Note that the number of subwindows evaluated by stereo and laser methods varies due to the changing depths in the scene. The lower plot reports the relative number of operations performed by the detector for each variation of our method compared to the standard OpenCV implementation. In each plot, the boxes denote the lower quartile, median, and upper quartile values, and the whiskers mark the full extent of the data.

We evaluated our implementation on a set of images, stereo depth maps, and laser range data captured by a B21r mobile robot equipped with a 640x480 stereo camera and a SICK PLS laser range-finder. The image set consists of 300 images containing a total of 416 labeled faces with a variety of lighting conditions and natural background clutter.

In the following sections, we present results for four different algorithms. The first version uses only the camera calibration and position information to compute geometric constraints and does not make use of any external depth information. The second extends this with the use of stereo depth, and the third uses data from the laser range-finder. Finally, as a control, we show the results of the unmodified OpenCV implementation.

For the control, we ran the detector at 25 scales, starting at 20 pixels and scaling by a factor 1.1, up to a maximum of 200 pixels across. We scanned the detector over all image positions, shifting the window in increments of $[0.5 \cdot \frac{w}{w_0}]$,

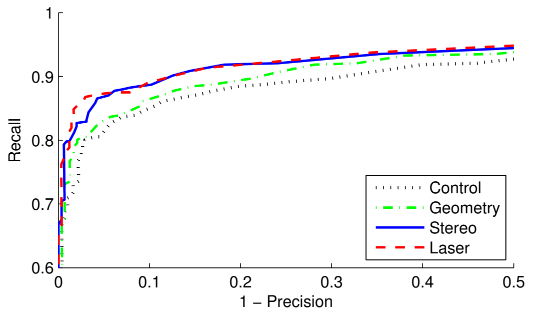

Fig. 4. Recall-precision curves comparing our approach to the baseline OpenCV face detector. The black dotted line shows the results of the standard detector; the green dotted/dashed line shows the results when geometric constraints are applied but no additional depth information is used; the blue line shows the results when using stereo depth; and the red dashed line shows the results when incorporating laser range data.

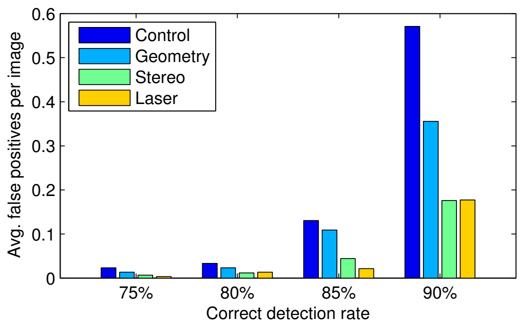

Fig. 5. The use of geometric constraints and external depth information results in a dramatic reduction in false positives. For comparison, we fix the accuracy and report the average number of false positives for each method. Using either stereo or laser depth information approximately 3 times fewer false positives are reported.

where w_0 and w are the initial and current window sizes, respectively, and [ ] denotes rounding. In all other experiments we start by computing the feasibility of each of these subwindows and only run the detector on the valid subset. To compute the geometric constraints, we used the following values (in meters): $h_{min} = 1.5$, $h_{max} = 2.0$, $s_{min} = 0.05$, and $s_{max} = 0.17$.

A. Computation

To provide a measure of the computational costs of each method, we present the total number of subwindows evaluated by the detector per image, as well as the the relative number of operations that were performed by the detector.

Figure 3 shows the relative cost per image for each method evaluated. Even without additional depth information, the use of geometric information cuts the computational costs in half. The addition of laser or stereo depth information reduces the computational costs by 85 percent.

There are additional costs involved in this technique beyond the cascade of detectors that we must also address. The most important is the calculation of the feasible depth ranges for each subwindow. Note that these ranges only need to be recalculated when the camera changes its tilt or roll. Additionally, if the camera remains horizontal (no roll), then two look-up tables containing the depth constraints for each value of v and w, respectively, can be used instead of computing the depth constraints for every (u, v, w). Thus, this cost is minimal.

When using laser depth readings, there is also the associated cost of the distance calculations. The gap-closing operation is the primary cost, but it only runs over the n laser readings (where typically $90 \leq n \leq 360$, depending on the laser device). This cost is negligible compared to the calculation performed by the detector.

A final possible cost is the stereo depth map calculation. Stereo calculations are usually quite costly to compute, but many sensors now perform the calculations in hardware. In addition, depending on the application, the depth map may already be available, as other concurrently running applications may require stereo vision. For the purposes of this work we do not consider the cost of stereo vision calculations in our analysis.

B. Accuracy

In addition to the reduction in required computation, we also see significant improvements in the face detection accuracy. In figure 4, we present the accuracy of each method in the form of a recall-precision curve. Recall is defined as the ratio of true positives returned to the total number of positives in the database. Precision is defined as the number of true positives found over the total number of positives returned by the detector (both true and false).

Because the face detector returns multiple positives for any given face (rectangular regions with slightly different positions and scale), a final stage used by the OpenCV implementation of Viola-Jones is a threshold on the number of detections which must overlap to be considered a detection. By lowering this threshold, the detector is more likely to correctly identify more faces in the scene (better recall), but this comes at the cost of a greater number of false positives (reduced precision). Likewise, raising this threshold will eliminate false positives but will also result in missed detections. Figure 4 shows the recall-precision curves generated by varying this threshold for each of the four methods.

As shown in figure 4, the use of geometric information alone results only in a slight improvement in accuracy. This is because fewer subwindows are discarded at this stage, and those that are tend to contain views of the floor and ceiling, where there is little texture. Thus, there is less of a chance that the standard detection algorithm would have misclassified these subwindows as a face.

However, the introduction of explicit depth information (either stereo or laser) results in a dramatic improvement. For instance, as shown in figure 5, the standard detector is able to correctly identify 90 percent of the faces in the dataset with an average of 0.57 false positives per image. In contrast, our algorithm is able to achieve the same detection

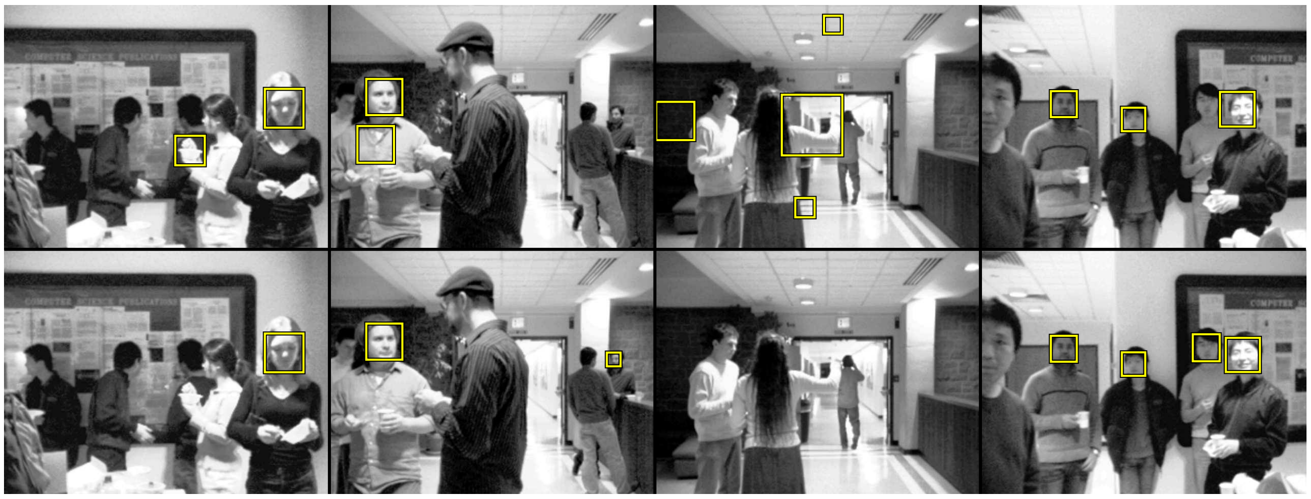

Fig. 6. Face detection results of the baseline detector (above) and laser-based method (below) are shown for four example images. For comparison, the threshold of each method was set so that each would have an accuracy of 90 percent. Notice that our algorithm is able to correctly identify some of the harder-to-find faces while discarding a number of false positives.

rate using either stereo or laser depth with three times fewer false positives.

V. CONCLUSION

This paper illustrates that simple modifications allow geometric constraints to be integrated with existing algorithms for face detection. In natural robot configurations, this offers dramatic performance gains. These gains are possible for a wide variety of object detection tasks beyond face detection.

There are two clear specific directions of future work. First, in this paper, we used hard constraints on valid regions for faces to appear. Probabilistic constraints on the feasible positions and sizes in 3D space could be used instead, allowing the detector to test regions in order of likelihood. This would also be an important part of integration into a tracking system, where temporal information could be integrated directly with the likelihood values. Second, faces are not the only objects which fit into this framework. We plan to apply this framework to other types of object detection which could make use of geometric and structural constraints. In particular, we intend to use these methods in the domain of car detection for an autonomous vehicle. Automobiles have similar geometric constraints which can be used to greatly reduce the search space in the image.

Applying generic vision algorithms directly to mobile robots with no modification fails to take advantage of vital information. For many algorithms, it is possible to exploit additional constraints such as environmental structure and sensors, resulting in significant improvements in performance. In general, we believe that robot vision tasks should be approached with the aim of using all relevant spatial information and fusing data from multiple sources.

REFERENCES

[1] H.A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:23–38, 1998.

[2] Haiyuan Wu, Qian Chen, and Masahiko Yachida. Face detection from color images using a fuzzy pattern matching method. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(6):557–563, 1999.

[3] Taigun Lee, Sung-Kee Park, and Mignon Park. A new facial features and face detection method for human-robot interaction. In *Proceedings of ICRA-05, The IEEE International Conference on Robotics and Automation (ICRA '05)*, pages 2063–2068, Barcelona, Spain, April 2005.

[4] Paul A. Viola and Michael J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[5] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *Proceedings of the IEEE International Conference on Image Processing (ICIP '02)*, volume 1, pages 900–903, Rochester, New York, September 2002.

[6] S.H. Kim, N.K. Kim, S.C. Ahn, and H.G. Kim. Object oriented face detection using range and color information. In *FG '98: Proceedings of the 3rd. International Conference on Face and Gesture Recognition*, page 76, Washington, DC, USA, 1998. IEEE Computer Society.

[7] Zachary Byers, Michael Dixon, Kevin Goodier, Cindy M. Grimm, and William D. Smart. An autonomous robot photographer. In *Proceedings of the IEEE/RSJ International Conference on Robots and Systems (IROS 2003)*, volume 3, pages 2636–2641, October 2003.

[8] M. Kleinehagenbrock, S. Lang, J. Fritsch, F. Lomker, G.A. Fink, and G. Sagerer. Person tracking with a mobile robot based on multi-modal anchoring. In *Proceedings of IEEE Workshop on Robot and Human Interactive Communication*, pages 423–429, 2002.

[9] J. Blanco, W. Burgard, R. Sanz, and J.L. Fernandez. Fast face detection for mobile robots by integrating laser range data with vision. In *Proc. of the International Conference on Advanced Robotics (ICAR)*, pages 953–958, 2003.

[10] J. Fritsch, M. Kleinehagenbrock, S. Lang, G.A. Fink, and G. Sagerer. Audiovisual person tracking with a mobile robot. In F. Groen, N. Amato, A. Bonarini, E. Yoshida, and B. Kröse, editors, *Proc. Int. Conf. on Intelligent Autonomous Systems*, pages 898–906, Amsterdam, March 2004. IOS Press.

[11] N. Bellotto and H. Hu. Multisensor integration for human-robot interaction. *The IEEE Journal of Intelligent Cybernetic Systems*, 1, July 2005.

[12] www.intel.com/research/mrl/research/opencv/.