An Autonomous Robot Photographer

Zachary Byers Michael Dixon Kevin Goodier Cindy M. Grimm William D. Smart
Department of Computer Science and Engineering
Washington University in St. Louis
One Brookings Drive
St. Louis, MO 63130
United States
{zcb1,msd2,bkg2,cmg,wds}@cse.wustl.edu

***Abstract*— We describe a complete, end-to-end system for taking well-composed photographs using a mobile robot. The general scenario is a reception, or other event, where people are roaming around talking to each other. The robot serves as an "event photographer", roaming around the same space as the participants, periodically taking photographs. These images are then sent to a workstation where participants can print the photographs out, or email them.**

I. Introduction

We describe a complete, end-to-end system for taking well-composed photographs using a mobile robot. The general scenario is a reception, or other event, where people are roaming around talking to each other. The robot serves as an "event photographer", roaming around the same space as the participants, periodically taking photographs. These images are then sent to a workstation where participants can print the photographs out, or email them.

The photography application was developed as a framework for investigating algorithms for standard robot tasks, such as navigation and localization, as well as novel human-robot interaction methodologies. The system we describe below is a baseline from which we can explore more advanced algorithms. A major feature of the photography application is that we have, in essence, a "sliding scale" for evaluation of the system — how many "good" versus "bad" pictures did the robot take. While not an objective scale, this evaluation, plus observation of the system in action, can provide a qualitative measure of how changing a component affects the entire system.

The current system is entirely event-driven. The robot searches for a potentially "good" photograph location, navigates there (if possible), identifies a good composition, and adjusts the camera until a "good" composition is found. At this point the robot takes a picture and begins the cycle again. Since people move, the robot must constantly evaluate its current state and determine if, for example, it needs to look for another photograph location because there are no people visible from its current location. Currently, our navigation is mostly reactive, although we have the ability to "tether" the robot to a visual landmark. Complete details of system's software architecture and basic navigation routines are beyond the scope of this paper, but can be found in the paper by Byers *et al* [1]. For more information on the quality of interaction between the robot and humans, see the paper by Smart *et al* [2].

Fig. 1. The robot and cameras.

II. The Robot

The system is implemented on an iRobot B21r mobile robot, with two cameras, mounted on a Directed Perception pan/tilt unit (see figure 1). All computation and control is performed on-board using a Pentium-III 800MHz processor, with 128MB of RAM. Other than the cameras, the only sensor used for the work reported here is the laser range-finder. This sensor returns 180 radial distance measurements, covering the front 180° of the robot, at a height of approximately 40cm from the floor.

The photography system uses a digital video camera to detect and frame subjects, and a digital still camera to actually take the final pictures. Both of these are mounted on the pan/tilt unit. The video camera (the lower one in figure 1) is a Sony DFW-VL500 IEEE 1994 (FireWire) camera, with a resolution of 640 by 480 pixels, capable of 30 frames per second. This camera operates continuously, with its output being used to detect possible subjects (see section IV).

Once a photograph is correctly set up, the other camera is used to take the picture. This camera is a Kodak DC290 digital still camera, and has a resolution of 1600 by 1200 pixels. It is connected to the robot through a USB interface and is capable of taking a flash photograph once every 20 to 30 seconds. The photographs are stored on the camera, and then downloaded to the robot in batch mode once every few minutes.

For this project, the system also includes a standard workstation, connected to the robot through a wireless ethernet link.

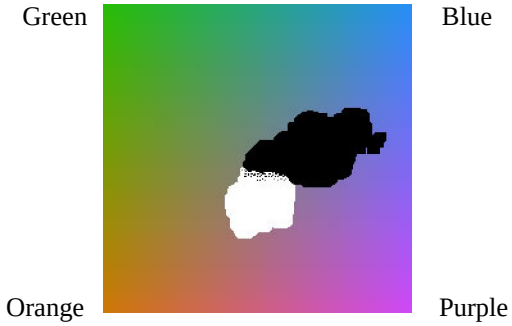


Fig. 2. All possible UV colors in a sample image. The colors in the image that correspond to skin are marked as white in the UV map, all other colors as black. Colors not marked were not identified in the training images. (This figure will render as uniform grey on a black and white printer.)

This workstation is used to display the photographs that the robot takes. Participants use a GUI to browse the photographs, email them, or print them out as souvenirs.

III. Previous Work

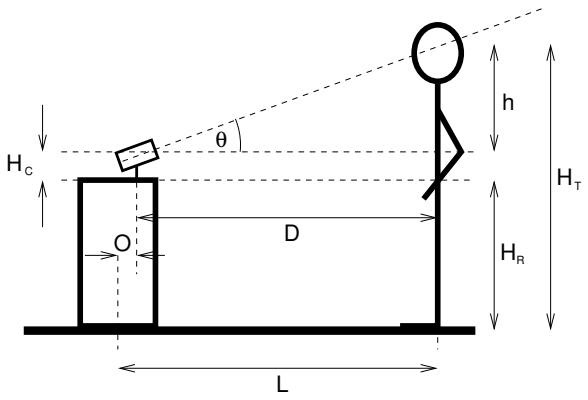
We are not aware of any previous work in the area of mobile robot photography. However, there are bodies of related work in the areas of image composition and face detection.

The majority of composition-related work appears in virtual worlds, and uses composition rules to position a virtual camera [3], [4], [5], [6], choose a location for a virtual cinematographer [7], [8], [9], [10], or control how a scene changes [11]. In a virtual world the system has the advantage of knowing exactly where all of the individuals are, so framing a shot is a well-defined optimization problem. Also, the camera is free to move anywhere, unlike a physical camera mounted on a robot. Many of the on-line systems also employ strategies for dealing with a constantly changing environment.

Rules from cinematography and TV interviews have also been used to control cameras in video teleconferencing [4]. There is also a growing area of research that uses a tight visual feedback loop to control robotic manipulators [12]. We share the feedback nature of these systems, but we differ in the required accuracy and on the fixed location of the camera. We do not need to position the robot or the camera with a great deal of accuracy. Our feedback loop is also operating in a relative coordinate system (the robot's position) without any knowledge of an absolute coordinate system.

Face detection and tracking have been widely studied. Approaches include simple skin detection [13], [14], learning from examples [15], [16], Eigenspace approaches [17], and template matching [18]. One of the best current approaches is the system by Viola and Jones [19], [20], which uses feature detection combined with fast feature comparison and discard to quickly find objects in an image.

Somewhat close in spirit to our approach is the work by Fleck *et al* [21], which identifies patches of skin in an image, and uses heuristics about the structure of the scene to relate the skin patches to human bodies.



$$h = D \tan\theta$$

$$D = L - O$$

$$H_T = H_R + H_C + D \tan\theta$$

Fig. 3. Calculating heights and distances using the laser range-finder.

IV. Finding Subjects

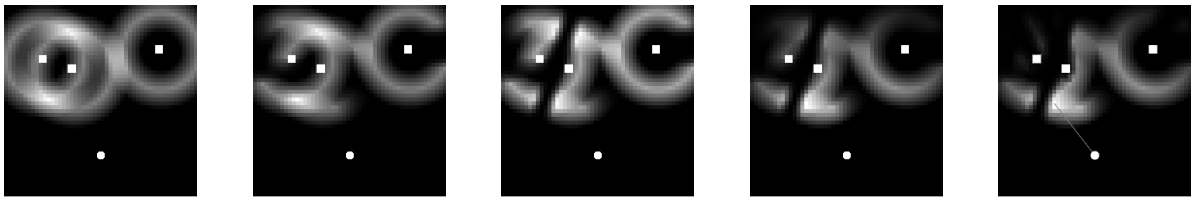
The first step in the photography process is to identify potential subjects. Since we are interested in taking photographs of people, this problem can be reduced to finding faces, and the approximate location of the associated person with respect to the robot. We assume that most of our subjects will be standing and will be adults. Currently, we are not detecting or using any information about the direction in which people are facing.

The face detection algorithm first finds all skin-colored blobs in an image. It then relates these blobs to readings from the laser range-finder to calculate the position and size of likely faces. Skin blobs that are the correct shape, size, and height from the ground are classified as faces. We discuss each of these steps in detail below.

A. Skin detection

The first step is to find skin-colored pixels in the image. It turns out that skin, even skin from different races, clusters tightly in many color spaces [13]. It has been shown that, for the type of application that we are interested in, the actual choice of color space has little impact on the performance of classification [22]. For this work, we use the UV plane of the YUV color space, which is a format our camera supports. We should reiterate at this point that we do not need perfect skin detection for our application, since we can heuristically remove false positives using data from other sensors. Using the YUV color space allows us to be fast, but without incurring many false negatives (skin patches not being identified in an image).

Although much of the previous work in identifying faces in an image has superior performance to our approach, it is also significantly more computationally expensive, and often makes assumptions about forward-facing faces. For example, the work by Viola and Jones [19] would require most of our



(a) (b) (c) (d) (e)

Fig. 4. Constructing the objective function to take into account (a) distance, (b) occlusion, (c) bisection, (d) movement, and (e) reachability. Lighter shades represent larger values of the objective function. The lowest white dot represents the robot position. The other dots are detected people.

processor resources in order to run at a reasonable rate. While it can be argued that this is not a limitation (since we could buy a faster processor), it does consume power. We are interested in long deployments of the system, and reduced processor usage equates directly to longer battery life. It is also not clear what the benefits of having a better face detection algorithm would be, and if they would justify the increased computational cost. This is something that we are currently evaluating.

We currently need to train the skin detection algorithm for every new environment that the system operates in. We do this by taking a small number of images, typically five to ten, and annotating them by hand, using a simple graphical interface. For all pixels identified as skin in the GUI, the corresponding areas in the UV plane are labelled as "skin". We similarly identify all pixels that are "not-skin". Once this initial assignment is done, we locally blur the regions in the table, and expand them a little. This has the effect of removing noise and making the regions more contiguous. Empirically, we have found that this leads to more robust skin detection. Figure 2 shows the portion of UV space that is classified as skin for a particular set of training images.

After the system is trained, we build a lookup table, indexed by U and V values, and use this to classify every pixel in an incoming image as "skin" or "not-skin". These classifications are then grouped together into blobs, and labelled as a potential face. At this point, we can throw out any potential faces that do not have a reasonable aspect ratio, since faces are generally taller than they are wide.

B. Range detection

The laser range-finder returns 180 distance readings approximately one degree apart over the front 180° of the robot. The camera pan/tilt unit is mounted in a fixed position on top of the robot. Given a pan angle we can determine which laser reading corresponds to a given pixel of the image, using simple geometry.

The laser range-finder is not mounted directly under the camera, and this offset, O, must be accounted for by subtracting it from the reading from the range-finder, L. The height of the robot, H_R, and the height of the camera, H_C, are constants. We assume that the floor is a level plane, and that the target is standing more-or-less upright.

Once we have identified the corresponding laser range-finder readings for each candidate face, we can apply our remaining heuristics. We assume that people are standing and are between four and seven feet tall. We also assume that all faces lie within a certain size range. Based on the distance returned by the range-finder, we can calculate the height and actual size of the skin patches corresponding to the candidate faces, as shown in Figure 3. Any candidates that are outside of the height or size limits are eliminated.

V. Navigation

Once we have identified potential subjects, we use simple rules from cinematography to pick desirable photograph locations. We favor locations where the robot is approximately five feet from the subject. We do not want to take pictures where one person is occluding another. We do not want to take pictures directly down the perpendicular bisector of two people, but would rather take pictures "over" the shoulder.

To calculate the best possible position we construct an objective function, which represents the expected quality of a picture taken from any given point. The space around the robot, which contains the people that we want to photograph, is discretized into a grid. In each grid cell we store a number corresponding to how good we think a picture would be from that point. Our composition rules are written in terms of these values, so it is easy to experiment with different rules and rule combinations.

A. Creating the objective function

The objective function encompasses the robot's current position and the positions of people it has found. The function is initially zero everywhere, and is updated as follows.

Distance from subject (Figure 4(a)). The ideal operating distance of the still digital camera's zoom and flash is between four and seven feet. Therefore, the robot should be in this range for at least one of the subjects. We increase the values of the objective function in a band around each subject, with the value peaking at a distance of 5.5 feet.

Occlusion (Figure 4(b)). Locations where faces appear to overlap will not yield good photographs. For each pair of subjects, we calculate the line that runs through them, and reduce the value of the cells along that line. Cells that lie on the line segment between the two subjects, however, are left unchanged.

Bisector (Figure 4(c)). Photos taken from along a perpendicular bisector between two subjects will result in both subjects being the same distance from the camera. If they are talking to each other, this results in two profile shots, which we would

(a) (b)

Fig. 5. Framing an image. (a) Optimal framing (large box) of the two detected faces (small boxes). (b) Optimal framing, with the rule-of-thirds lines.

like to avoid. We calculate the perpendicular bisector of all subjects within five feet of each other and decrease the values of the objective function along this line.

Movement (Figure 4(d)). We want to minimize the distance the robot travels to reduce the possibility that a photo opportunity will disappear. We also wish to discourage the robot from remaining at the same location for multiple photographs. We decrease the objective function to zero for all points closer than 2 feet from the robot, or further than 20 feet. Between these extremes, we decrease the values linearly based on distance.

Reachability (Figure 4(e)). To make navigation simple, we avoid destinations which would require sophisticated path planning and obstacle avoidance. A point is considered unreachable if the robot cannot drive in a straight line to it without going through something. We use the laser rangefinder information to calculate the horizon of all reachable points, and we set the objective function to zero for all points beyond this horizon.

Once the objective function is constructed, we simply look for the point with the greatest value and drive towards it. If an obstacle is encountered on the way, the destination is recalculated based on the new obstacles and the current location of faces.

VI. Photograph Composition

Once the robot has identified some possible subjects for the photograph, and has moved into an appropriate position, it must determine a good composition for the picture. Simply taking a photograph with people in it is unlikely to result in a "good" picture; there are some basic composition rules that must be followed. In this section, we describe the composition rules that we use, and how they affect the pan, tilt and zoom of the camera as we take photographs.

A. Composition Rules

We use four well-accepted rules from photography: the rule-of-thirds, the empty space rule, the no-middle rule, and the edge rule [23]. The use of these rules is illustrated in figure 5.

Rule-of-thirds: It is best to place the faces in a photograph at, or near, the one-third and two-thirds horizontal lines in an image.

Empty space: The faces in an image should occupy at least the middle third of the image, either horizontally or vertically.

No-middle: Do not place a single subject exactly on the midline of the photograph.

Edge: Subjects should not be placed at, or crossing, the edge of the photograph.

B. Grouping Subjects

Once a set of candidate subjects has been identified, using the techniques described in section IV, we must determine what subset of these candidates to photograph, and how to best frame the shot, according to the above rules.

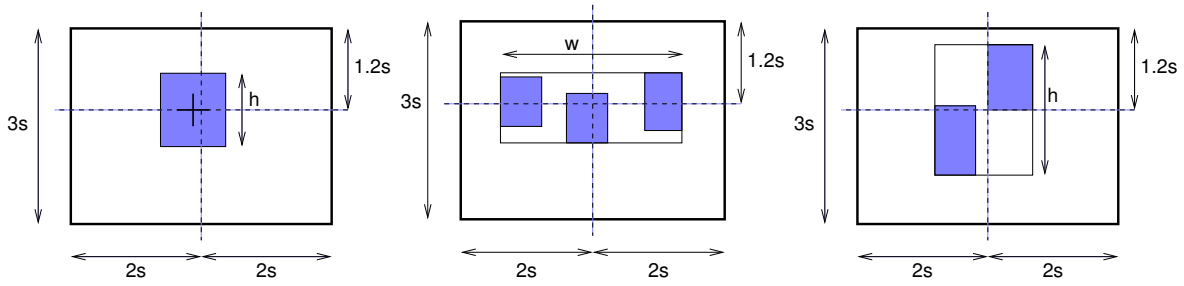
We use an iterative procedure to determine the subset of candidates to include in the photograph. We begin by considering only the center-most candidate in the image. This candidate is framed using the single person framing rules, described below. If the framed area includes any new candidates, it is expended, using the group framing rules. This procedure iterates until the framing for the current subset does not cut across any other candidate faces. The two framing algorithms that we use for this procedure are as follows.

Single person: Candidate subsets with only one face require a tighter framing than those with groups of people. The rules applied here are the no-middle rule, the empty space rule, and the rule-of-thirds. The ideal framing is calculated by placing the face slightly to the left or right of the vertical center line, and ensuring that it takes up two-thirds of the image height (see Figure 6(a)). The center of the face is positioned slightly below one-third down the image. This takes into account people's hair and necks, which typically extends beyond the bounding box (which is based on skin-colored regions).

Groups of people: The rules applied here are the rule-of-thirds and the empty-space rule. The ideal framing is found from the width of the enclosing box for all of the faces. Again, the centerline of this box is conservatively placed slightly below one-third down the image, but is now centered in the image, as is shown in Figure 6(b). Wide groups of faces, where $w > 1.6h$, are dealt with differently than narrow ones. This threshold, and the ratios in Figure 6, were determined empirically to result in pleasing compositions.

C. Taking the Shot

Once we have established the ideal framing for a particular shot, we pan, tilt and zoom the camera to achieve this framing.



(a) $s = 1.5h$ (b) $s = \frac{3}{8}w$ (c) $s = \frac{1}{2}h$

Fig. 6. Calculating the ideal framing for (a) a single face, and (b) a wide group of faces, and (c) a narrow group of faces.

This control is done in a closed-loop fashion, with the ideal framing constantly being compared to the current framing. This allows much more reactivity in the case where subjects are not completely static, and are likely to move as the shot is being composed.

The closed-loop control of the framing does introduce a potential problem, however. We have noticed that the robot sometimes has trouble converging on the correct framing, and can oscillate back and forth for a relatively long period of time. To counteract this problem, we have introduced a notion of timeliness to the system. It is more important to take a reasonable picture quickly than it is to take a perfect one slowly. We have implemented a decaying threshold for how close the actual framing has to be to the ideal framing for a picture to be taken. As the robot spends more and more time trying to get the ideal framing, the less accurate the actual framing has to be to be considered good enough. Empirically, we have found that this removes the seeking problems around the perfect framing, without significant loss of quality in the resulting pictures.

VII. Results

We have deployed the robot photographer at several real-world events, including a major computer graphics conference and a wedding reception. In this section, we describe the results of some of these deployments, and attempt to measure the success of the photography system.

The most significant deployment to date was at the SIGGRAPH 2003, as part of the Emerging Technologies exhibit. The robot ran for 40 hours, over a period of five days, and interacted with over 5,000 people. Other significant deployments are described in the paper by Byers *et al* [1].

There are no hard success metrics for this application. One possible way to evaluate performance is to see how many of the photographs that were taken are (subjectively) considered to be "good". At SIGGRAPH, the robot took 3,008 photographs. Of these, 1,053 (35%) were either printed out, or emailed to someone. This means that slightly over one-third of the photographs that the robot took could be considered good enough for someone to want to keep them. We believe that this compares favorably to average human performance.

Rating	Percentage
Very Bad	18%
Bad	25%
Neutral	28%
Good	20%
Very Good	9%

TABLE I
Evaluation of the photographs taken.

In order to get a more reliable estimate of the quality of the photographs, we have also run a set of evaluation experiments. Subjects, who were not associated with the project, were asked to rate sets of randomly-selected photographs taken by the robot. Each photograph was rated as either "very bad", "bad", "neutral", "good", or "very good". Over 2,000 photographs were evaluated, with the results shown in table I. Roughly one third of the photographs were evaluated as being "good" or "very good". While this does not sound like a very large number, we believe it is very encouraging, especially given the simplicity of our algorithms. We are aiming at an "amateur snapshot" level of photography, and even professional photographers discard a large number of the photographs that they take. A selection of the rated pictures, taken from each of the five categories, is shown in figure 7.

VIII. Conclusions and Future Work

We have demonstrated a baseline system that is capable of navigating through crowds taking "snapshot" quality pictures. Using a number of straightforward algorithms, combined with data from more than one sensor, and a knowledge of the task and environment, we have achieved a surprising performance level.

We believe that there are many avenues for continued research within this application. Our future research will use the framework of the robot photographer to investigate more general problems in mobile robotics, including the following.
Navigation and localization: Can we use information about where people are to build more robust behaviors, especially in crowded environments?
Advanced subject-finding: Can we incorporate more advanced face-finding routines or stereo vision to derive better

(a) Very bad

(b) Bad

(c) Neutral

(d) Good

(e) Very good

Fig. 7. Example photographs.

information about where people are? Can we use this information to build maps of likely photograph positions, and then use these to inform the navigation routines?

Human-robot interaction: What is the best model for robot-human interaction in this environment? Do we need to know more about human state (such as, are they waving to get the robot's attention) and if so, how do we capture it? How can we most easily externalize the robot's state (or what we want people to believe the robot's state to be) to human observers?

ACKNOWLEDGMENTS

We would like to thank Jacob Cynamon, Patrick Vaillancourt, Michal Bryc, and Robert Pless for their help with various aspects of the project. This work was supported in part by NSF REU award #0139576, and by NSF award #0196213.

REFERENCES

[1] Z. Byers, M. Dixon, W. D. Smart, and C. M. Grimm, "Say cheese!: Experiences with a robot photographer," in *Proceedings of The Fifteenth Innovative Applications of Artificial Intelligence Conference (IAAI-03)*, Acapulco, Mexico, August 2003.

[2] W. D. Smart, C. M. Grimm, M. Dixon, and Z. Byers, "(not) interacting with a robot photographer," in *Proceedings of the AAAI Sping Symposium on Human Interaction with Autonomous Systems in Complex Environments*, Stanford, CA, March 2003.

[3] B. Gooch, E. Reinhard, C. Moulding, and P. Shirley, "Artistic composition for image creation," *Eurographics Workshop on Rendering*, 2001.

[4] W. H. Bares and J. C. Lester, "Cinematographic user models for automated realtime camera control in dynamic 3D environments," in *Proceedings of the Sixth International Conference on User Modeling*, 1997, pp. 215–230.

[5] S. Fleishman, D. Cohen-Or, and D. Lischinski, "Automatic camera placement for image-based modeling," *Computer Graphics Forum*, vol. 19, no. 2, pp. 101–110, 2000.

[6] É. Marchand and N. Courty, "Image-based virtual camera motion strategies," in *Graphics Interface*, 2000, pp. 69–76.

[7] L. He, M. F. Cohen, and D. H. Salesin, "The virtual cinematographer: A paradigm for automatic real-time camera control and directing," *Computer Graphics*, vol. 30, no. Annual Conference Series, pp. 217–224, 1996.

[8] S. M. Drucker, T. A. Galyean, and D. Zeltzer, "Cinema: A system for procedural camera movements," in *1992 Symposium on Interactive 3D Graphics*, vol. 25, March 1992, pp. 67–70.

[9] S. M. Drucker and D. Zeltzer, "Camdroid: A system for implementing intelligent camera control," in *1995 Symposium on Interactive 3D Graphics*. ACM SIGGRAPH, April 1995, pp. 139–144.

[10] B. Tomlinson, B. Blumberg, and D. Nain, "Expressive autonomous cinematography for interactive virtual environments," in *Proceedings of the Fourth International Conference on Autonomous Agents*, C. Sierra, M. Gini, and J. S. Rosenschein, Eds. Barcelona, Catalonia, Spain: ACM Press, 2000, pp. 317–324.

[11] M. Kowalski, J. Hughes, C. Rubin, and J. Ohya, "User-guided composition effects for art-based rendering," *ACM Symposium on Interactive 3D Graphics*, 2001.

[12] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automattion*, vol. 12, no. 5, pp. 651–670, 1996.

[13] J. Yang, W. Lu, and A. Waibel, "Skin-color modeling and adaptation," in *ACCV (2)*, 1998, pp. 687–694.

[14] K. Schwerdt and J. Crowley, "Robust face tracking using color," in *Proc. of 4th International Conference on Automatic Face and Gesture Recognition*, 2000, pp. 90–95.

[15] K. K. Sung and T. Poggio, "Example-based learning for view-based human face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 39–51, 1998.

[16] H. A. Rowley, S. Baluja, and T. Kanade, "Human face detection in visual scenes," in *Advances in Neural Information Processing Systems*, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, Eds., vol. 8. The MIT Press, 1996, pp. 875–881.

[17] A. Pentland, B. Moghaddam, and T. Starner, "View-based and modular eigenspaces for face recognition," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'94)*, Seattle, WA, June 1994.

[18] N. Oliver, A. Pentland, and F. Berard, "Lafter: Lips and face real time tracker," in *Proc. Computer Vision and Patt. Recog.*, 1997.

[19] P. Viola and M. Jones, "Robust real-time object detection," in *Proc. of IEEE workshop on Statistical and Computational Theories of Vision*, Vancouver, Canada, July 2001.

[20] ——, "Rapid object detection using a boosted cascade of simple features," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, HI, December 2001.

[21] D. Forsyth and M. Fleck, "Automatic detection of human nudes," *International Journal of Computer Vision*, vol. 32, no. 1, pp. 63–77, 1999.

[22] M. C. Shin, K. I. Chang, and L. V. Tsap, "Does colorspace transformation make any difference on skin detection?" in *Proceedings of the IEEE Workshop on Applications of Computer Vision*, Orlando, FL, December 2002.

[23] T. Grill and M. Scanlon, *Photographics Composition*. American Photographics Book Publishing, 1990.