

A Vision System for Personal Service Robots: Resilient Detection and Tracking of People

Alexandre R.J. François and Gérard G. Medioni
Institute for Robotics and Intelligent Systems
University of Southern California
afrancoi,medioni@usc.edu

June 2006

Abstract

This report describes the design and implementation of a computer vision system that performs real-time people detection and tracking from stereo color video. The system, developed in the context of a project that aims to empower personal robots with advanced vision capabilities, illustrates a systematic approach to combining efficient but brittle algorithms into robust systems. The system's tracking core builds upon established monocular algorithms for face detection and color-based tracking to perform stereo multi-target people (head) tracking. These algorithms interact in the dynamic system according to a general tracking pattern whose core is a feedback loop. Systems designed according to this pattern exhibit resilience properties, such as bootstrapping capability, robustness to input data noise, adaptability to changing environment. As a stand-alone integrated demonstration, the system incorporates video capture, processing and result visualization for evaluation purposes. It was designed using the Software Architecture for Immersipresence model. Parallel implementations on both Windows and Linux platforms, using the open source MFSM architectural middleware and the Intel OpenCV library, reliably detect and track people at interactive rates.

1 Introduction

This report describes the design and implementation of STEVI v.1, a demonstration computer vision subsystem that performs real-time people detection and tracking from stereo color video. STEVI v.1, developed in the context of a project that aims to empower personal service robots with advanced vision capabilities, illustrates a systematic approach to combining efficient algorithms into robust systems.

1.1 Motivation

Personal service robots, whose purpose is to “educate, entertain, or assist, or protect in the home” (as defined by *Robotic Trends* magazine) are attracting significant interest in both the research and commercial communities. Such robots can be especially useful for example in assisting people with disabilities and in addressing the needs of the aging population in industrial societies. A service robot must autonomously interact with its environment in a naturalistic manner [16], and in particular be able to interact and communicate with ordinary people in a natural and social way [9, 5, 8]. It is generally recognized that robots designed to interact with humans, in environments designed by and for humans, must have human-oriented perception [9]. In order for a personal service robot to function, it must have some awareness of its environment. In particular, it is critical for the robot to know whether, when and where people are present. STEVI v.1 addresses this particular capability.

Vision for robots encompasses most of the difficult challenges while adding some specific ones. The vision subsystem must operate in concert with the overall (dynamic) robotic system, at rates that are consistent with human expectations. Furthermore, mobile robots must operate robustly in unstructured environments. Consequently, efficient but brittle vision algorithms cannot individually provide satisfactory solutions.



Figure 1: Composite images of tracking data overlaid on left and right input color images. Note the head area size consistent with the person’s distance to the cameras (the further away, the smaller) and the consistent labels in the left and right views.

1.2 Overview of Approach

The key to building robust and efficient computer vision systems consists in *designing* complex, integrated systems whose overall properties go beyond the sum of that of their individual algorithms. STEVI v.1’s tracking core builds upon established monocular algorithms for face detection and color-based tracking to perform stereo multi-target people (head) tracking. A classifier-based face detection algorithm based on [21] performs initial target acquisition. Each detected face area instantiates a target candidate to which is attached a color model (histogram) initialized from the face area in each image of the stereo pair. A mean shift-based algorithm [6] tracks the targets (or rather their color models) that have sufficient combined support in both images.

Interaction between data and processing elements implementing these algorithms in the system are captured in the system’s architectural specification, and can be abstracted into *architectural patterns*. STEVI v.1’s tracking core is a feedback loop. If all tracking algorithms are feedback loops by nature, the static formulation and sequential implementation of tracking systems often hide or abstract their dynamic properties. For example, the fundamental nature of tracking as feedback is implicit in iterative methods such as mean shift-based tracking [11]. Explicitly modeling the dynamic aspect of tracking allows to address dynamic properties such as bootstrapping, robustness to input data noise, adaptability to changing environment. Making explicit the dynamic nature of tracking systems also allows appropriate and efficient application of traditional static algorithms to dynamic data.

Tracking as feedback is not a new concept. For example, Jebara and Pentland describe a 3D adaptive feedback tracking system for faces [15]. The system switches between two modes: robust face detection and fast tracking. It is designed as two feedback loops operating at very different rates to accommodate the complexity of the algorithms involved and the computing power available on contemporary hardware. STEVI v.1 illustrates a general feed-back pattern that presents an integrated approach to detection and tracking. This pattern is not specific of the particular algorithms used. It is, by design, open to the parallel operation of multiple algorithms (e.g. for detection) and provides a formal framework for implementing fusion algorithms.

Specific integration of stereo processing and color-based tracking has been explored before. For example, Moreno et al. describe a real-time system that performs 3D head tracking [20]. Depth information computed from stereo scales the sampling area for mean shift tracking. STEVI v.1’s system never computes pixel-level depth information from stereo, but rather uses stereo information to estimate the depth of higher-level objects (e.g. faces). This efficient use of stereo information increases the robustness of the system through redundancy and enhanced adaptivity.

As a stand-alone integrated demonstration, the STEVI v.1 system incorporates video capture, processing and result visualization for evaluation purposes. Figure 1 shows composite images of tracking information overlaid on the input left and right images. The target labels are consistent in left and right views (and

across time, not visible in the figure).

1.3 System design

Usage of a relevant formalism greatly facilitates the design, study and implementation of complex dynamic systems. This project adopted the Software Architecture for Immersipresence (SAI) framework [13].

SAI specifies a formal architectural style [12] whose underlying extensible data model and hybrid (shared memory and message-passing) distributed asynchronous parallel processing model facilitates the specification of asynchronous data stream processing software systems. A graph-based notation for architectural designs allows intuitive system representation at the conceptual and logical levels, while at the same time mapping closely to the physical level.

Figure 2 shows the conceptual specification graph for STEVI v.1 in the SAI notation. In the SAI model, *volatile data* flows down streams (notation: thin lines) defined by connections between processing centers called *cells* (notation: squares), in a message passing fashion. Repositories called *sources* (notation: circles) hold persistent data to which connected cells (connection notation: thick lines) have shared concurrent access. The directionality of stream connection between cells expresses dependency relationships. Dashed boxes in the figure identify three main groups of components: (1) The *Stereo input and pre-processing* subsystem regroups the multiple camera video input and subsequent pre-processing of the video frames flowing on the stream; (2) the *Stereo multi-target detection and tracking* subsystem is explained in detail in the remainder of this document; (3) the *Visualization* subsystem provides a visual insight into the tracker's state for debugging and demonstration purposes. The design is inherently parallel, and the modularity of the SAI style facilitates design evolution.

Note that the graph presents an intuitive view of the system. Because the visual language in which it is expressed has well defined semantics, the graph is also a formal specification (in this case at the conceptual level).

The next four sections of this report explain in details the workings of the stereo multi-target tracking subsystem: Section 2 introduces the principles of the feedback loop tracking pattern; Section 3 presents the stereo mean shift-based tracking process; Section 4 describes the stereo target acquisition process; Section 5 addresses target prediction and tracker update mechanisms. Section 6 reviews system implementation and performance. Finally, Section 7 presents a summary of contributions and directions for future work.

2 Integrated Detection and Tracking

Tracking is inherently a dynamic process. Making explicit the persistence of target trackers and the volatile nature of target observations from data measurements naturally reveals the underlying feedback process.

2.1 A general tracking pattern

The subgraph delimited by the box labeled *Stereo multi-target detection and tracking* in Figure 2 illustrates a general tracking pattern that captures the essence of the feedback approach. The *Tracking source* holds a list of target trackers. At each moment in time, the state of these trackers represents the latest known persistent information for processing. A feedback loop anchored by the *Predict targets* cell and the *Update trackers* cell accesses and maintains this information. The acquisition of each new data sample (stereo pair) triggers a cycle of the loop: The prediction process (*Predict targets* cell) generates a list of expected observations based on the current trackers. A tracking subsystem (*Track targets* cell) uses predictions and input data to produce a list of confirmed tracked target observations. A residual analysis subsystem (*Acquire targets* cell) explains input data not accounted for by the predictions, resulting in initial detection of additional targets. Finally, an updating process (*Update trackers* cell) integrates confirmed target observations into the corresponding trackers, and create new trackers for newly detected targets.

2.2 Properties of the pattern

This general pattern is inherently robust to some issues that are difficult to handle in static or sequential approaches to online video tracking. Depending on specific design choices, a system based on this pattern

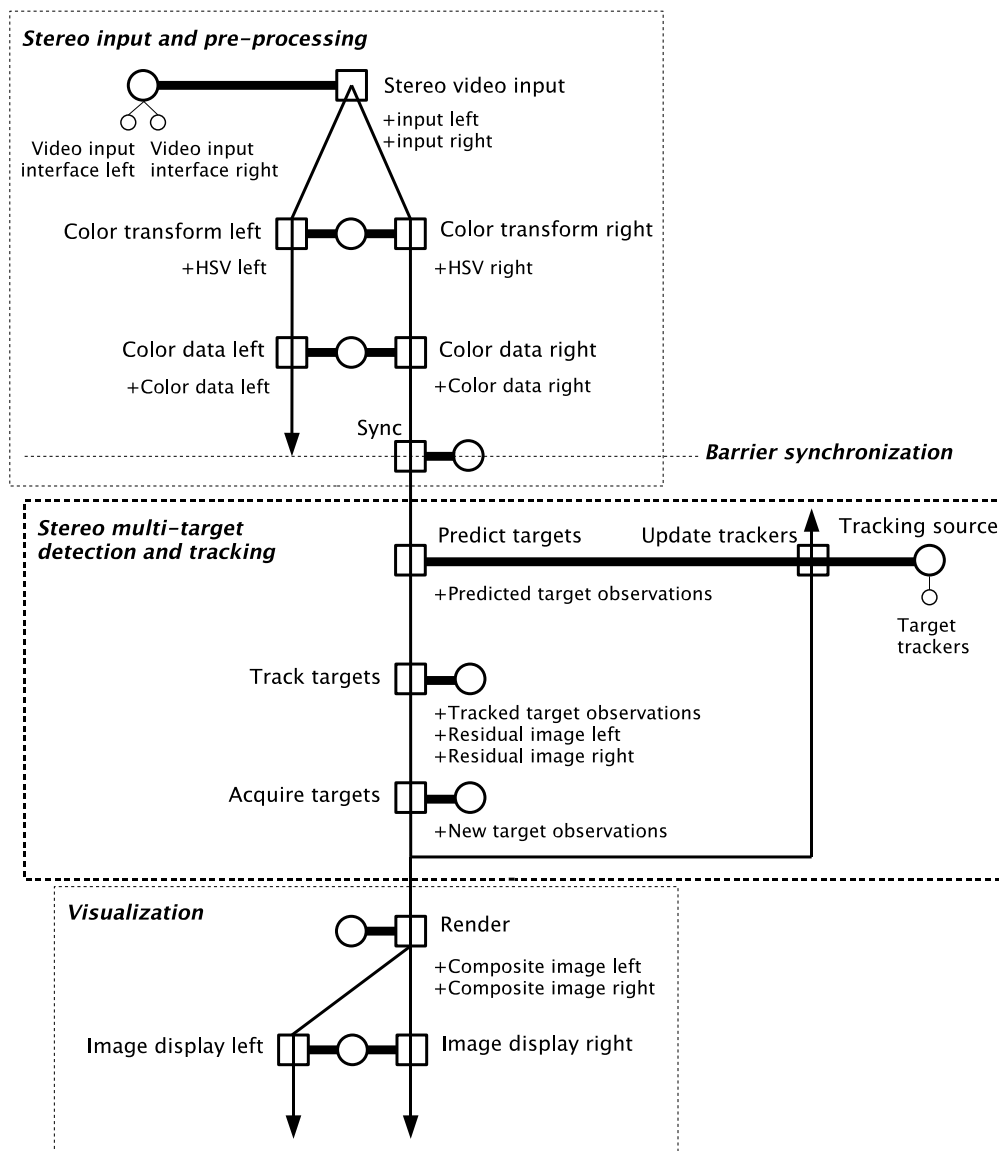


Figure 2: STEVI v.1 conceptual level system architecture in SAI notation.

may exhibit properties that are out of the reach of the individual algorithms involved for detection and tracking. Because the Tracking and Acquisition subsystems operate on the same control loop, the tracking process is naturally *bootstrapped*: unknown targets are automatically acquired. Furthermore, the absence of observation for a given known target in a given input sample need not necessarily result in the loss of the corresponding tracker (depending on the update policy). The system is thus resilient to a wide range of disruptions and discontinuities in the input signal as well as errors in intermediate results caused by failures of the individual algorithms.

2.3 Application in STEVI v.1

STEVI v.1 specifically addresses real-time multiple people detection and tracking in stereo input video. Therefore the tracking and acquisition subsystems rely on algorithms specialized in face detection and skin color tracking, with emphasis on the stereo integration, as described in the next sections. The algorithms selected do not put any constraint on the motion of the targets or camera set.

The next three sections address respectively the tracking of known targets, new targets acquisition, and target prediction and tracker update.

3 Target tracking

Given a set of predicted observations (generated from the trackers) and some input data, the tracking process produces a set of corresponding target observations that will be used to update the state of their respective trackers. A variety of approaches can be applied to this task. In STEVI v.1, a mean shift-based approach continuously adjusts the model to match the data. Stereo information is used to increase the robustness of the system (cross-checking) and to dynamically adjust some parameters of the mean-shift.

3.1 Mean shift-based color tracking

The mean shift algorithm [6] is a robust non-parametric iterative gradient descent technique for finding the mode of probability distributions. Convergence towards the mode of a probability distribution is achieved by starting from a reasonable position, and iteratively shifting to a position computed as the weighted mean of a kernel centered at the current position. This algorithm exhibits, under reasonable conditions, very attractive convergence and robustness properties. In monocular mean shift-based color tracking [7], a target is characterized by a color model (histogram), initialized from an external process, and its last known position. Each input frame is converted into a probability distribution by attributing to each color pixel its value in the target's histogram. Starting from the last known position, mean shift is iterated to find the mode of the distribution, estimated target location in the new frame. The search area can be restricted around the last known position of the target, resulting in possibly large computational savings. The size of the area to use for sampling the distribution is a fixed parameter. The Continuously Adaptive Mean SHIFT (CAMSHIFT) algorithm [3] extends mean shift-based color tracking by continuously computing the size and orientation of the tracked area, which is fed back to the mean shift process as the size of the sampling area in the next frame. However, CAMSHIFT was not designed for, and does not perform well in situations involving long range interaction and multiple targets. Instead, STEVI v.1's tracking algorithm uses stereo information computed for higher level structures (regions rather than pixels) to adjust the size of the mean shift sampling window in successive frames.

3.2 Stereo mean shift-based tracking in STEVI v.1

The input to the *Track target* cell consists of left and right input color data information and a set of predicted target observations. The output of the mean shift tracking is a set of tracked target observations, i.e. target observations updated according to the data using the predictions as starting point. In addition, the cell produces left and right residual images, obtained from the original left and right input images, that are used to reduce the search space for the target acquisition subsystem.

Each target prediction comprises two sets (left and right) of the following: color model (histogram), current position and size. The tracking produces for each confirmed target an observation that comprises

new positions and sizes in the left and right images. For each target, only a fixed number of iterations is performed on each data sample. Typically, only one position update should be necessary, although a low input sample rate relative to target image velocity might require more iterations. A key aspect is that mean shift is never applied until convergence (i.e. iterated an undetermined number of times) for a given stereo image pair. The feedback loop in the tracking subsystem ensures that the new target positions are used as starting point of their respective shifts in the next input samples. Note that if the input data is constant, the tracked positions will converge to the same positions as they would in the traditional off-line iterative approach.

Use of stereo redundancy confers added robustness to data noise and better adaptivity of the mean shift process. It is important to emphasize that the tracking process does not involve any matching between left and right information. A target observation explicitly encodes the conceptual relationship between its left and right observation models (area and color histogram), which are enforced during tracking. Because the cameras are close to each other and looking in parallel directions, the face area is constrained to have the same size in both views. Furthermore, the shift observed in the right and left images should be similar (although not strictly equal). Finally, the Euclidian distance between the centers of the tracked regions in the left and right images estimates the disparity of the target. The size of the target area is computed as a function of the disparity (from stereo calibration data), and thus indirectly fed back to the mean shift process.

Figure 1 shows the tracked positions of three targets in both the left and right images. The apparent size of the face area is estimated using stereo information. Color-based tracking is robust to head orientation (as long as enough skin color remains visible).

4 Target acquisition

The target acquisition process analyzes input data not already explained by the tracking process. STEVI v.1 relies on a face detection module to acquire people's heads. Although new target acquisition is a sparse event, the need to continue searching for new targets in residual data (i.e. regions of the input images not accounted for by the tracking module), as well as timely integration of new targets in the tracking process, put efficiency constraints on this part of the system as well.

4.1 Face detection algorithm

The literature contains many approaches to face detection ([22, 14] offer recent surveys). In general, appearance-based and learning-based techniques have produced the best results and show the most promise to date [17]. Among these, the approach originally proposed by Viola and Jones [21] is the most compatible with the real-time requirements of robotic vision. The recognition engine is a cascade of weak classifiers utilizing Haar wavelet-like features. AdaBoost learning improves performance and efficiency of the learning process and of the resulting classifiers. This approach (with various subsequent extensions by other authors [18, 19]) is suitable for real-time performance, and was thus adopted for face detection in STEVI v.1.

4.2 Stereo face detection in STEVI v.1

The *Acquire targets* cell combines the results of monocular face detection in left and right residual images to infer the presence of new targets in the scene. The residual images, produced by the *Track target* cell, invalidates images areas where faces are already being tracked. Cross validation in left and right views increases the system's robustness to false positives in the monocular face detection process. Simultaneous detection of overlapping face areas in left and right images results in the instantiation of a target observation structure, with initial color histograms, positions and sizes in each view. The target area size is forced to be identical in both views, consistently with a similar constraint enforced in the tracking. Performing target detection for each input pair reduces the overall impact of detection failure in a particular input.

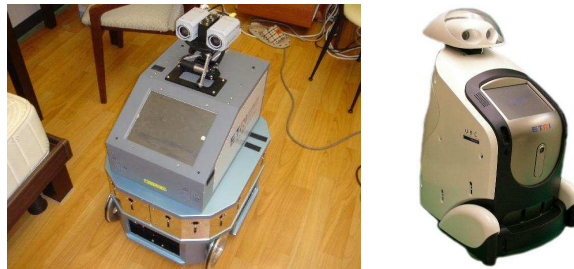


Figure 3: ETRI robot platform: current prototype Weber-N (left) and consumer product design (right). (Pictures courtesy of ETRI)

5 Target prediction and tracker update

The target prediction and tracker updating processes anchor the tracking feedback loop. Together with appropriate target trackers structures, they could implement a variety of traditional methods, such as Kalman filtering [2] and particle filtering [1]. Because STEVI v.1's detection and tracking subsystem operates in 2D image space and under the assumption of a sufficient input frame rate, simplicity and efficiency are preferred to complex algorithms. The prediction process transfers the last known characteristics for each known target. Similarly, the updating process is kept to the simplest minimum set of operations, by transferring the observed characteristics of observed targets, creating new trackers for newly acquired targets, and removing the trackers not supported by observation. This last policy means that the tracker for a target not observed in a single input sample will be removed, and the target's identity lost. A lost target will be re-acquired as soon as the acquisition algorithm will allow, but will be assigned a new tracker (with a different label). If one considers the output of the system to be the individual labeled trajectories for tracked targets (this information is captured in the list of target tracker states at each moment), these trajectories would be fragmented in the presence of occlusions or change in appearance. The implementation of more elaborate prediction and update algorithms is one possible way to address some of these issues by allowing target trackers to persist through accidental absence of observation, thus by making the tracking process locally more robust.

6 Implementation and performance

STEVI v.1 was implemented in C++ using the Modular Flow Scheduling Middleware (MFSM) [10], an open source architectural middleware that provides code support for SAI's architectural abstractions. The graph specification of Figure 2 constitutes a conceptual-level map of the code. Systems designed with SAI and implemented with MFSM are automatically multi-threaded, and can directly take advantage of multiple CPUs, hyper-threading and multi-core processor architectures. The MFSM project comprises a number of open source functional modules that allow to focus coding effort on project specific components of the system. For example, modules for video input, image display, and other general purpose modules were directly reused. The basic computer vision algorithms underlying the system are established in the literature. STEVI v.1's implementation encapsulates face detection and color-based tracking functionalities provided by the Intel Open Source Computer Vision Library (OpenCV) [4]. The initial STEVI v.1 system was developed on the MS Windows platform. The same system was ported to Linux with minimal effort, thanks to the cross-platform MFSM code and module base, and the availability of Intel's OpenCV for both platforms. Furthermore, because of the modularity of the model, the development of subsequent evolutions of STEVI will directly benefit from the work done on v.1.

The running systems reliably detect and track people at approximately 15 frames-per-second on contemporary hardware. Video input (images 320x240 pixels) is acquired by two color cameras (USB or Firewire) mounted on a calibrated stereo rig. STEVI v.1 systems have run continuously for hours. A version using a generic classifier cascade trained on frontal faces (as shipped with the MS Windows version of OpenCV v.3.1) detects most new targets as soon as the faces become visible at the resolution required by the face

detector. False positives, quite frequent with the monocular algorithm, are extremely rare occurrences. Once acquired, a face in motion can be tracked for several seconds to several minutes. Principal reasons for discontinuity are change of lighting/color (caused either by target motion in the environment or environmental change), self occlusion. These situations expose the limited robustness of the largely skin-based color model, and can be addressed by building more complex target models, or combining multiple detection algorithms, which would allow more robust tracking. Occlusion by environment obstacles or other targets also causes tracking discontinuity, due to the 2D nature of the detection data used in the tracking process. As already mentioned, more elaborate prediction and update algorithms could make the tracking process locally more robust. Another promising approach to improving overall system robustness is the addition of a higher level 3D tracking subsystem fed by the current detection and tracking system.

In all experiments, the most computationally expensive task is face detection, already performed on reduced-resolution images for efficiency purposes. The visualization subsystem, useful only for demonstration purposes, also consumes a non-negligible part of computation power. STEVI v.1 was integrated and tested on the Weber-N personal service robotic platform developed by the Electronics and Telecommunications Research Institute (ETRI, see Figure 3).

7 Conclusion

This report described the design and implementation of STEVI v.1, a computer vision system that performs real-time people detection and tracking from stereo color video. STEVI v.1 illustrates the integration of established monocular algorithms for face detection and color-based tracking in a general dynamic tracking pattern whose core is a feedback loop. This pattern provides a general template for designing tracking systems that exhibit resilience properties such as bootstrapping capability, robustness to input data noise, adaptability to changing environment.

The complete stand-alone integrated system incorporates video capture, processing and result visualization for evaluation purposes. It was designed using the SAI model. Parallel implementations on both Windows and Linux platforms, using the open source MFSM architectural middleware, reliably detect and track people at interactive rates.

Subsequent versions of STEVI will achieve increased robustness through the use of more elaborate dynamic target models, and of multiple detection algorithms. Another promising approach to improving overall system robustness is the addition of a higher level 3D tracking subsystem fed by the current detection and tracking system.

7.1 Acknowledgments

This work was conducted in the context of a collaborative project with the Electronics and Telecommunications Research Institute (ETRI), a Korean non-profit, government-funded research organization.

References

- [1] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 2002.
- [2] Y. Bar-Shalom and T. E. Fortmann. *Tracking and data association*. Academic Press, 1988.
- [3] Gary R. Bradski. Computer video face tracking for use in a perceptual user interface. *Intel Technology Journal*, Q2 1998.
- [4] Gary R. Bradski. The OpenCV Library. *Dr. Dobb's Software Tools for the Professional Programmer*, November 2000.
- [5] Cynthia Breazeal. Socially intelligent robots. *ACM Interactions*, 12(2):19–22, 2005.
- [6] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, May 2002.

- [7] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *Proceedings of the Int. Conf. Computer Vision and Pattern Recognition*, pages II: 142–149, 2000.
- [8] David J. Feil-Seifer and Maja J. Matarić. Socially assistive robotics. In *Proceedings of the International Conference on Rehabilitation Robotics*, Chicago, IL, July 2005.
- [9] T. Fong, I. Nourbakhsh, and K. Dautenhahn. A survey of socially interactive robots. *Robotics and Autonomous Systems, Special issue on Socially Interactive Robots*, 42(3-4):143–166, 2003.
- [10] Alexandre R.J. François. Modular Flow Scheduling Middleware. *mfsm.SourceForge.net*.
- [11] Alexandre R.J. François. CAMSHIFT tracker design experiments with Intel OpenCV and SAI. Technical Report IRIS-04-423, Institute for Robotics and Intelligent Systems, University of Southern California, July 2004.
- [12] Alexandre R.J. François. A hybrid architectural style for distributed parallel processing of generic data streams. In *Proceedings of the International Conference on Software Engineering*, pages 367–376, Edinburgh, Scotland, UK, May 2004.
- [13] Alexandre R.J. François. Software Architecture for Computer Vision. In G. Medioni and S.B. Kang, editors, *Emerging Topics in Computer Vision*, pages 585–654. Prentice Hall, Upper Saddle River, NJ, 2004.
- [14] E. Hjelmås and B.K. Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83(3):236–274, September 2001.
- [15] Tony S. Jebara and Alex Pentland. Parameterized structure from motion for 3d adaptive feedback tracking of faces. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 144–150, 1997.
- [16] Kazuhiko Kawamura, Robert T. Pack, and Moenes Iskarous. Design philosophy for service robots. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics: 'Intelligent Systems for the 21st Century'*, volume 4, pages 3736–3741, Vancouver, BC, Canada, October 1995.
- [17] Stan Z. Li and Juwei Lu. Face detection, alignment, and recognition. In G. Medioni and S.B. Kang, editors, *Emerging Topics in Computer Vision*, pages 385–455. Prentice Hall, Upper Saddle River, NJ, 2004.
- [18] S.Z. Li, L. Zhu, Z. Zhang, A. Blake, H. Zhang, and H. Shum. Statistical learning of multi-view face detection. In *Proceedings of the European Conference on Computer Vision*, page IV:67 ff., 2002.
- [19] R. Lienhart, A. Kuranov, and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In Bernd Michaelis and Gerald Krell, editors, *Proceedings of the 25th DAGM Symposium on Pattern Recognition*, pages 297–304, Magdeburg, Germany, September 2003. Springer Verlag.
- [20] Francesc Moreno, Adria Tarrida, Juan Andrade-Cetto, and Alberto Sanfelin. 3d real-time head tracking fusing color histograms and stereovision. In *Proceedings of the International Conference on Pattern Recognition*, pages I: 368–371, 2002.
- [21] Paul Viola and Michael J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.
- [22] M.H. Yang, D.J. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, January 2002.