Distributed Barcode Tracking in Visual Sensor Networks

Leander Hendrikx
Vrije Universiteit Brussel - imec
ETRO Dept., Brussels, Belgium
leanderhendrikx@gmail.com

Eli De Poorter
Ghent University - imec
IDLab, Ghent, Belgium
Eli.DePoorter@UGent.be

Adrian Munteanu
Vrije Universiteit Brussel - imec
ETRO Dept., Brussels, Belgium
acmuntea@etrovub.be

***Abstract*—A novel approach for robot tracking and identification based on barcodes is proposed in this paper. The proposed system tracks robots fitted with barcodes identifying them. The system performs distributed visual processing and collaborative barcode tracking, whereby the nodes exchange processed visual information with appropriate neighboring nodes, informing them about incoming targets. The proposed tracking system is deployed on a visual sensor network (VSN) based on the Raspberry Pi. Its practical realization includes various components that handle real-time collaborative tracking, streaming, communication, configuration, management, monitoring, and deployment. The proposed system is able to efficiently and accurately track barcodes in real time. Complexity modeling demonstrates that the proposed distributed system brings substantial complexity reductions compared to a system of independently operating cameras. Experimental results demonstrate that the proposed system can track multiple barcodes in real time with an accuracy of less than 1 centimeter.**

***Keywords*—Visual Sensor Networks, Distributed processing, Robot tracking, Barcode detection.**

I. Introduction

Nowadays, one-dimensional barcodes are globally used for tagging products. This work makes use of barcodes to localize, identify and track robots in visual sensor networks. Barcode detection is a well-researched area, many approaches being based on mathematical morphology. The approach of [1] uses edge detection combined with filtering and morphological operations to detect barcodes. The method in [2] uses a black top-hat transform and a filter for high-density regions. The work in [3] uses a blob detector to identify potential barcodes. The blobs are then filtered based on their shape and converted to a feature space before feature clustering is applied. Recent advances include [4] which combines [1] and [2].

Evolving from barcode detection in still pictures towards barcode detection and tracking in video calls for specific algorithmic designs. Barcode detection and tracking in video has recently been proposed in [5]. The method in [5] indicates that robust robot tracking and identification based on one-dimensional barcodes is feasible. However, [5] suffers from extremely low frame rates (less than one frame per second on average) when barcode tracking is deployed on low-power embedded devices. The distributed barcode tracking system presented in this work advances over our past developments in [5] by bringing algorithmic changes and system design optimizations in order to achieve both robustness and real-time barcode tracking of multiple targets with a low-power VSN.

II. Proposed Distributed Visual Sensor Architecture

The basic idea behind the proposed distributed tracking system is that the nodes in the VSN can go into sleep mode when no barcodes are present in their field of view. The neighboring nodes will wake up a node whenever a barcode is likely to enter that node's field of view. That saves energy compared to a VSN of independently operating cameras whereby each node searches for barcodes, tracks and decodes them at all times.

The barcode tracking method is divided into two algorithms. The first algorithm is responsible for searching an incoming barcode. The search is performed in a predefined area which is determined based on the information received from the barcode trackers running on the neighboring cameras in the VSN. The second algorithm is responsible for tracking a barcode. First, the algorithm predicts the location of the tracked barcode. Then, it attempts to localize the barcode in an area defined around the predicted location. In a subsequent step, if the barcode is successfully localized, the algorithm extracts the barcode from the image, attempts to decode it, and checks if the decoded identifier matches with the identifier from the past. If the barcode was not localized or if the decoding is unsuccessful, the location gets discarded.

The barcode localization algorithm consists of two steps: barcode detection and location refinement. Barcode detection is inspired by [4]. The algorithm performs the following steps: (i) black top-hat transform, (ii) low-intensity thresholding to remove areas that are not barcodes, (iii) Otsu thresholding, and (iv) morphological filtering. In contrast to [4], the proposed method, which uses an additional thresholding step, does not perform contrast stretching and adopts a different method to determine the barcode's bounding box and its orientation.

The second step refines the location of the detected barcodes by applying corner detection combined with robust bounding-box detection. The process is illustrated in Figure 1. First, the contour of the detected barcode (smallest white rectangle in Figure 1a) is determined. Then, the bounding box of that contour is calculated and scaled up (Figure 1b). Lastly, the four corners of the barcode are located using Harris corner detection (Figure 1c).

The barcode extraction algorithm serves two purposes, namely, (i) determining if the input image contains a barcode or not, and (ii) extracting barcodes from the image. The algorithm exploits the strong directional features of barcodes.

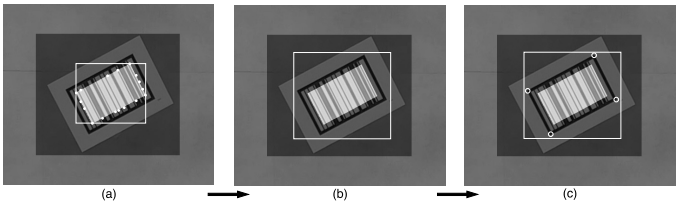

Fig. 1. Location refinement algorithm.

The algorithm uses the four points provided by the localization algorithm to extract the barcode from the image and to warp it prior to decoding. The result is a rectified image of the barcode. The algorithm subsequently extracts a set of five sample lines in both horizontal and vertical directions. One of the sets should have a high number of edges, whereas the other set should have (almost) no edges. If that is not the case, the location gets discarded.

The decoding algorithm assigns the barcode patterns to the correct characters. The decoding process also returns a confidence figure that represents how confident the decoder is about the decoded identifier. That figure is based on how many of the bars inside the patterns were classified correctly.

III. System Overview and Implementation Challenges

The proposed system is composed of two entities: the nodes in the VSN (based on the Raspberry Pi in our practical realization) and the central server. Figure 2 shows an overview of the components comprised by these entities and how they are connected. The main purpose of the central server is visualizing the processed information that is sent by the VSN. A screenshot of that visualization is shown in Figure 3. Figure 4 shows a picture of the physical setup.

The most important component in a node is the Barcode Tracker. The internal structure of that component is shown in Figure 5. The arrows indicate the direction in which information is sent between the different modules of the barcode tracker. The type of arrow indicates the type of data that is sent (dashed: frame data; full-line: barcode data). The numbers next to the arrows indicate the order in which the data is sent.

The first step is to capture a frame from the camera and to send that frame to the Tracker (arrow 1). The Tracker then creates Track and Search threads depending on the number of tracked barcodes and pending search requests (arrow 2). These threads execute the previously explained track and search algorithms. When these threads finish executing, they send the processed information back to the Tracker (arrow 3). The tracker then processes that information to see if there are outgoing barcodes. If so, the information is sent to the Search Manager (arrow 4) which wakes up the node that is likely to see the outgoing barcode. Next, the Tracker sends the frame data together with the processed information for that frame to the RTPStreamer (arrow 5). That data is then sent to the central server for further processing (visualization). The data from the different nodes is synchronized by the central server using timestamps. The system clocks of all nodes are synchronized with the system clock of the central server so that the maximum difference between any two nodes is 2ms.

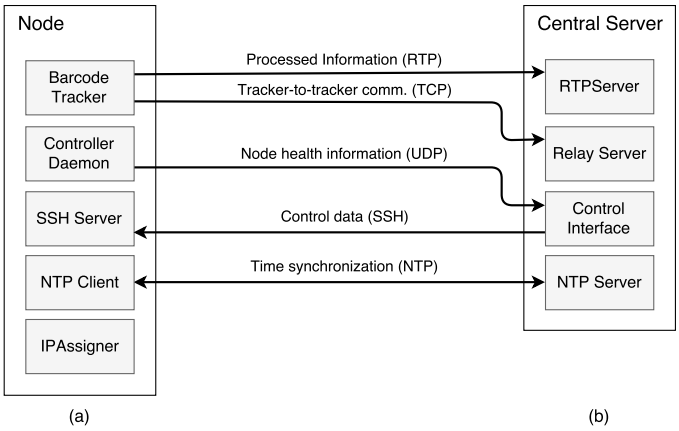

Fig. 2. Diagram showing the components in a node and in the central server.

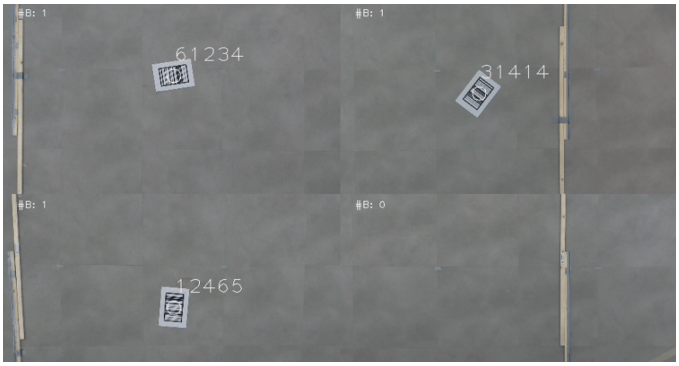

Fig. 3. Visualization of the processed information (server side).

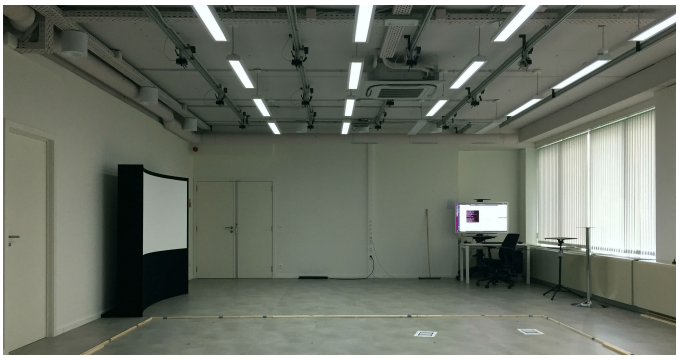

Fig. 4. Physical setup with cameras facing vertically downwards at a height of approximately 2.5m.

Implementing the proposed system on a low-power VSN has many challenges. The first and biggest challenge is to design the system in such a way that it is efficient, scalable and also flexible. The second challenge is the difficulty of testing and debugging such a system. The problem originates from the fact that it is impossible to store the captured frames while the tracker is running, as that would require an additional compression step. That means that there is no way of executing the tracker on exactly the same input. Using a static scene is not a solution either since the camera also captures noise. The only method for repeating results involves capturing and storing frames when the tracker is not running, and then using those frames as input for the tracker. Unfortunately, those

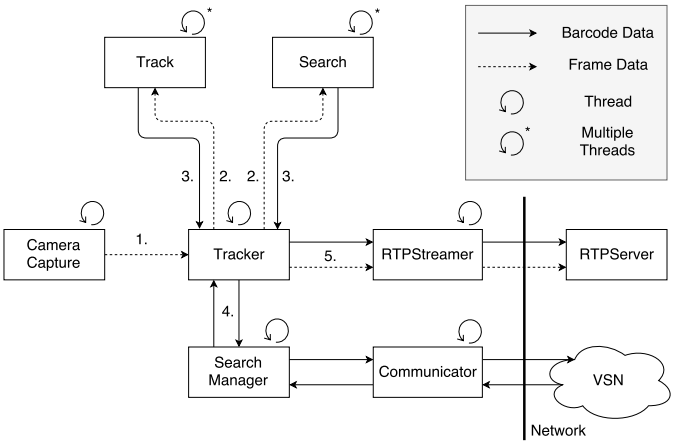

Fig. 5. Diagram showing the internal structure of the barcode tracker.

frames have to be compressed due to the limited size and write speed of the SD-cards on the nodes. Compression artefacts could influence the accuracy or robustness of the tracker.

IV. Experimental Results

The localization accuracy of the proposed system was evaluated. The accuracy was measured in real world coordinates with respect to a self-defined origin point outside the camera's field of view. The test showed a maximum difference of 0.9 cm between the measured and tracked positions. The area uniquely covered by one camera is about $3m^2$ (total coverage is about $4m^2$).

Next, a mathematical model for the execution time of the proposed tracking system was derived. Due to space limitations, the model cannot be detailed here. That complexity model was compared to a model for a non-distributed version of the system corresponding to barcode tracking performed by a network of independently operating cameras. The biggest difference between the two is that the non-distributed tracker has to search the entire frame whereas the distributed tracker can search in small pre-defined areas, determined based on information from neighboring nodes. A plot of both models is shown in Figure 6. The yellow line represents the non-distributed tracking model, the blue and red lines are the upper and lower bounds respectively of the distributed tracking model. Note that the models represent the execution time of a single barcode tracker executed on a single thread.

To compare the two versions in terms of computational efficiency, the gain of distributed relative to non-distributed tracking is calculated. That gain is calculated at system level and expressed in terms of the average number of barcodes per tracker. [6] is a white paper about Kiva Systems. They report values ranging from 0.0005 robots/m^2 to 0.065 robots/m^2, meaning that the proposed system could expect to see between 0.0015 and 0.195 robots per tracker. Figure 7 shows that the system-wide gain for these realistic figures lies somewhere between 40x and 270x.

V. Conclusions

This work proposes a distributed visual processing system to identify and track multiple barcodes in real time. Experiments have shown that the system has a high accuracy, of less than 1 cm for an area of $3m^2$. The proposed system is also much less complex than a non-distributed equivalent system, with complexity gains between 40x and 270x. The proposed system was practically realized on low-power hardware, which allows for saving power, installation and maintenance costs.

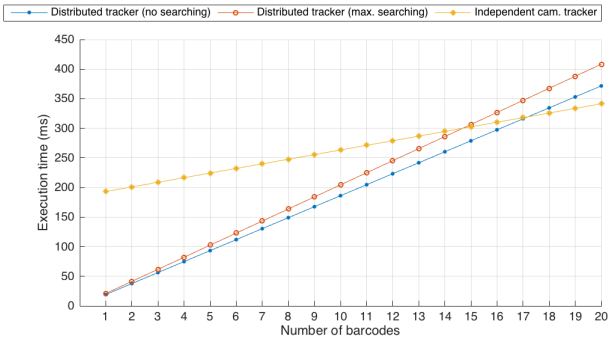

Fig. 6. Plot of the estimated execution time for the proposed and independent camera tracking systems.

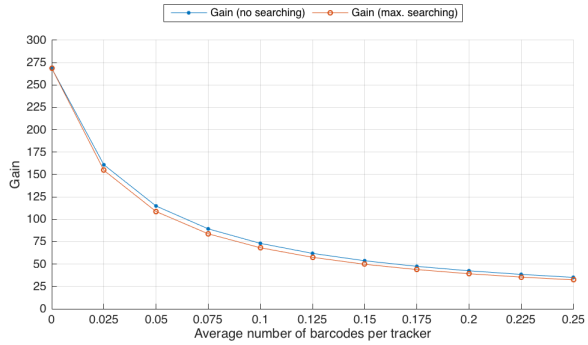

Fig. 7. Plot of the estimated system-wide gain by using the proposed distributed tracking system over an independent camera tracking system.

Acknowledgments

The research work was funded by the Fund for Scientific Research-Flanders (FWO-Flanders), projects G084117 and G025615.

References

[1] T. R. Tuinstra, "Reading barcodes from digital imagery," Ph.D. dissertation, PhD thesis, Cedarville University, 2006.

[2] X. J. Juett and X. Qi, "Barcode localization using bottom-hat filter," *NSF Research Experience for Undergraduates*, 2005.

[3] C. Creusot and A. Munawar, "Real-time barcode detection in the wild," in *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*. IEEE, 2015, pp. 239–245.

[4] S. Kaur and R. Maini, "Implementation of barcode localization technique using morphological operations," *International Journal of Computer Applications*, vol. 97, no. 13, 2014.

[5] E. Olti, T. Verbeke, G. Braeckman, V. T. Dadarlat, and A. Munteanu, "Robot tracking in low-power visual sensor networks," in *Proc. 10th Int. Conf. on Distributed Smart Cameras*. ACM, 2016, pp. 19–24.

[6] "Is Kiva Systems a Good Fit for Your Distribution Center? An Unbiased Distribution Consultant Evaluation." MWPVL, Tech. Rep., 2012. [Online]. Available: http://www.mwpvl.com/html/kiva_systems.html