Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Auto-organized visual perception using distributed camera network

Richard Chang*, Sio-Hoi Ieng, Ryad Benosman

Institut des Systemes Intelligents et de Robotique (ISIR-CNRS), 4 Place Jussieu, 75252 Paris Cedex 05, France

ARTICLE INFO

Article history:
Received 22 July 2008
Received in revised form
2 July 2009
Accepted 8 July 2009
Available online 23 July 2009

Keywords:
Camera network
Image coding
GMM
Information extraction
Dynamic organization

ABSTRACT

Camera networks are complex vision systems difficult to control if the number of sensors is getting higher. With classic approaches, each camera has to be calibrated and synchronized individually. These tasks are often troublesome because of spatial constraints, and mostly due to the amount of information that need to be processed. Cameras generally observe overlapping areas, leading to redundant information that are then acquired, transmitted, stored and then processed. We propose in this paper a method to segment, cluster and codify images acquired by cameras of a network. The images are decomposed sequentially into layers where redundant information are discarded. Without the need of any calibration operation, each sensor contributes to build a global representation of the entire network environment. The information sent by the network is then represented by a reduced and compact amount of data using a codification process. This framework allows structures to be retrieved and also the topology of the network. It can also provide the localization and trajectories of mobile objects. Experiments will present practical results in the case of a network containing 20 cameras observing a common scene.

1. Introduction

As cameras are becoming common in public areas they are a powerful information source. Camera networks have been intensively used in tracking or surveillance tasks [1,2]. Most multi-camera systems assume that the calibration and the pose of the cameras are known, standard networks applications also imply other highly constraining tasks such as: 3D reconstruction, frames synchronization, etc. Baker and Aloimonos [3], Han and Kanade [4] introduced pioneering approaches of calibration and 3D reconstruction from multiple views. The reader may refer to [5–7] for interesting works on camera networks and to [8–10] for synchronization issues. Another limitation is that the cameras must also be stationary, the field of view of the network is then rigidly set and cannot be adapted according to the events in the scene. Most of the applications implying that the use of a set of cameras are processing information by incrementing acquired data. Every single camera acts as an individual entity that does not necessarily interact with the other ones. Usually the camera transfers its information regardless to the behavior of the other ones. Thus, if the network is dense enough, obvious redundancies are unavoidable and resources like bandwidth, mass storage system are simply wasted. One can expect to overcome these problems by coordinating smartly the efforts of each camera relying on the main idea that they are forming a unique vision sensor. It is also unreasonable to use raw images, to avoid unnecessary data transfer within the network. Data compression methods preserving relevant information should then be used. Scenes can be described using their contents relying on lines and edges to build geometric models from images [11]. In other cases, visual features can be merged with other modalities such as ultrasound sensors [12] to introduce robustness. Several aspects of the environment can also be extracted from images like walls, doors and vacant spaces [13]. Recent works on bag-of-features [14] representations have become popular, as they introduce geometry-free features to characterize local subimage using statistical tools.

It is often constraining to use camera networks as the high number of sensors needs permanent external tuning usually performed by a human operator. The aim of this paper is to introduce a geometry-free method that allows camera networks systems to estimate their topology and auto-organize their own activities according to the content of the scene and the task to be achieved. The paper introduces a common description visual language used by all cameras to exchange information about scenes. A sampling method of acquired images into subimages combined with bag-of-feature allowing their codification is presented. In a second stage, a multi-layer data reduction architecture is introduced, it is inspired by the statistical organization of the human retina [15]. This convergent structure as will be seen allows to remove redundancies. Finally a functional layer gathers cameras as single visual entities performing identified tasks.

This paper is organized as follows: in the next section, the multi-layer coding is presented. Each transition from the lowest stage to the higher one is detailed. In the third section we show that geometric structures can be recovered from such coded camera

* Corresponding author.
E-mail address: richard.chang@lis.jussieu.fr (R. Chang).

0921-8890/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2009.07.008

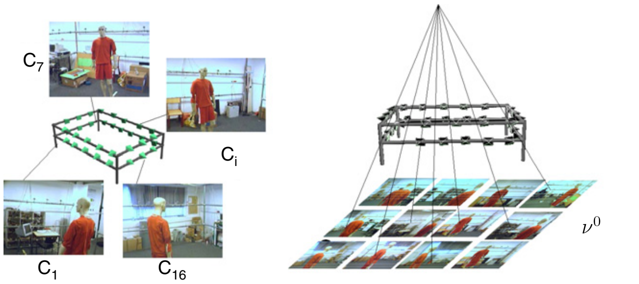

Fig. 1. Planar representation of the cameras' location in 3D onto a plane representing the first layer ν^0.

network: both cameras relative positions and scene object localization can be estimated up to some metric properties. In the last section, experimentations are tested on real images and results are provided.

2. Multi-layer image coding

To allow an easier handling of the camera network and the location of cameras, a planar topology of the network is introduced. As shown in Fig. 1 the relative position of the cameras are represented in a plane ν^0 set as the first layer. The multi-layer structure only need the coarse topology of the camera network (i.e. knowing which cameras are direct neighbors). We assumed in this section that this topology is known. The layer ν^0 deals with a planar representation of the camera network. Each camera is only placed relatively to its neighbors without any metric position.

In what follows ν^j is a plane at level j and ν^j_i will represent its ith element.

2.1. From acquired images to codified images (ν^0 to ν^1)

The goal of this section is to sample acquired images into representative patches. Each patch as will be seen will be compared to a codebook, and a codified image is produced. It is important to notice that the codebook is the same for all cameras, allowing further comparisons.

2.1.1. Characterizing texture

Texture can be measured using different approaches. In what follows we choose to use a measure similar to [16]. It relies on the computation of a histogram of the difference between the value of pixels of images. Given an image I, each value of its histogram of differences h_I is given by:

$$h_I(i) = \sum_{x,y,x',y' \in I}^{x \neq x' \vee y \neq y'} \text{diff}(I, x, y, x', y', i), \quad i \in [0, 255]$$

with

$$\text{diff}(I, x, y, x', y', i) = \begin{cases} 1 & \text{if } |I(x, y) - I(x', y')| = i \\ 0 & \text{else.} \end{cases}$$

In a second stage, the histogram h_I is normalized, to ensure an invariance according to the size of I.

2.1.2. Generating codebooks

Let $F_z(I)$ be a function allowing the decomposition of an image I into several textured patches:

$$F_z(I) = \{z_0, z_1, \dots, z_n\} \quad \text{with } I = \bigcup_{i=0}^{n} z_i.$$

Let $T = \{h_{z_0}, h_{z_1}, \dots h_{z_n}\}$ be the set containing all texture descriptors of patches z_i of I. The idea is to sample T to reduce the number of descriptors to $m \leq n$. We then add to T a metric function expressed by $\text{dist}(h_{z_i}, h_{z_j})$ and a reference texture patch h_{ref}. The reference patch is set to a patch containing a single color, corresponding to a uniform area. In a second stage all the representation of patches contained in T are compared to h_{ref} and sorted, from the less to the more textured. The set T_s corresponding to the ordered set T becomes:

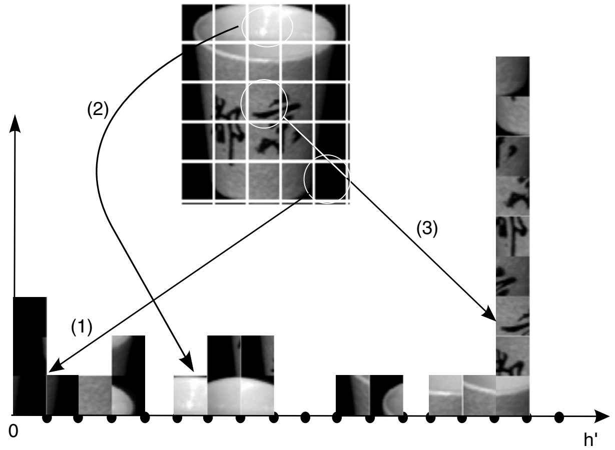

Fig. 2. Example of decomposing an image into patches. Extracted patches are compared to a uniform patch and sorted according to the complexity of their texture. The reference patch is set to 0 on axis h'. The mahalanobis distance is used as the metric of comparison. Example of patches from the less (1) textured to the more textured (3) are shown. The patch shown in (2) contains a light reflection that is assimilated to texture after normalization.

$$T_s = \{h_{\text{ref}}, h'_{z_0}, h'_{z_1}, \dots h'_{z_n}\}$$

with $\text{dist}(h_{\text{ref}}, h'_{z_i}) \leq \text{dist}(h_{\text{ref}}, h'_{z_j})$ if $i < j$.

An example of T_s is given by Fig. 2, where for a better understanding a simple object is considered.

The mahalanobis distance is used as a metric function and is set so for the rest of the paper. At this point, T_s is then sampled into m areas. For each area, only the median patch is selected. The resulting selection gives the codebook V:

$$V = \{h_{\text{ref}}, h'_{z_0}, h'_{z_1}, \dots h'_{z_m}\}, \quad V \subset T_s$$

that corresponds to the most representative patches.

2.1.3. Decomposing images into known patches of V

Let I_{acq} be an acquired image, I_{acq} is decomposed into z_{acq_i} patches. Each computed patch must be compared to the content of V, we then set a function Reco that transforms the patches of acquired images into patches of the codebook V:

$$\begin{aligned} \text{Reco} : \mathcal{P} \times \mathcal{C} &\rightarrow \mathcal{C} \\ (z_{acq_i}, V) &\longmapsto \text{Reco}(z_{acq_i}, V) = V \quad \text{if } h_{z_{acq_i}} \in V \\ &\qquad\qquad\qquad\quad = V \cup h_{z_{acq_i}} \quad \text{otherwise.} \end{aligned}$$

where $\mathcal{P}$ is the set of all patches, and $\mathcal{C}$ the set of all codebooks. In case a new patch is detected, it is added to the codebook as a new entry. The acquired image I_{acq} is then codified using the patches of the codebook, the resulting image I_{cod_i} given by:

$$I_{cod_i} \in \cup \text{Reco}(z_{acq_i}, V). \tag{1}$$

2.1.4. Optimal decomposition of images

An efficient decomposition must produce an optimal and possibly unique partitioning of images. In addition it would be interesting to produce less patches, but of variable size so that they can cover homogeneous texture zones.

In order to achieve an optimal generation of patches, a quadtree-like algorithm is set up. The quadtree algorithm cuts recursively images into subimages. Starting from the initial image,