Computer Vision and Image Understanding 114 (2010) 1180–1202

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

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

A fast stereo matching algorithm suitable for embedded real-time systems

Martin Humenberger [a,*], Christian Zinner [a], Michael Weber [a], Wilfried Kubinger [a], Markus Vincze [b]

[a] AIT Austrian Institute of Technology, Donau-City-Strasse 1, 1220 Vienna, Austria
[b] Automation and Control Institute (ACIN), Vienna University of Technology, Gusshausstrasse 27-29, 1040 Vienna, Austria

ARTICLE INFO

Article history:
Received 26 January 2009
Accepted 17 March 2010
Available online 4 May 2010

Keywords:
Stereo matching
Real-time stereo
Census
Embedded computer vision
DSP
GPU

ABSTRACT

In this paper, the challenge of fast stereo matching for embedded systems is tackled. Limited resources, e.g. memory and processing power, and most importantly real-time capability on embedded systems for robotic applications, do not permit the use of most sophisticated stereo matching approaches. The strengths and weaknesses of different matching approaches have been analyzed and a well-suited solution has been found in a Census-based stereo matching algorithm. The novelty of the algorithm used is the explicit adaption and optimization of the well-known Census transform in respect to embedded real-time systems in software. The most important change in comparison with the classic Census transform is the usage of a sparse Census mask which halves the processing time with nearly unchanged matching quality. This is due the fact that large sparse Census masks perform better than small dense masks with the same processing effort. The evidence of this assumption is given by the results of experiments with different mask sizes. Another contribution of this work is the presentation of a complete stereo matching system with its correlation-based core algorithm, the detailed analysis and evaluation of the results, and the optimized high speed realization on different embedded and PC platforms. The algorithm handles difficult areas for stereo matching, such as areas with low texture, very well in comparison to state-of-the-art real-time methods. It can successfully eliminate false positives to provide reliable 3D data. The system is robust, easy to parameterize and offers high flexibility. It also achieves high performance on several, including resource-limited, systems without losing the good quality of stereo matching. A detailed performance analysis of the algorithm is given for optimized reference implementations on various commercial of the shelf (COTS) platforms, e.g. a PC, a DSP and a GPU, reaching a frame rate of up to 75 fps for 640 × 480 images and 50 disparities. The matching quality and processing time is compared to other algorithms on the Middlebury stereo evaluation website reaching a middle quality and top performance rank. Additional evaluation is done by comparing the results with a very fast and well-known sum of absolute differences algorithm using several Middlebury datasets and real-world scenarios.

1. Introduction

For modern mobile robot platforms, dependable and embedded perception modules are important for successful autonomous operations like navigation, visual servoing, or grasping. Especially 3D information about the area around the robot is crucial for reliable operations in human environments. State-of-the-art sensors such as laser scanners or time-of-flight methods deliver 3D information, that is either rough or has low resolution with respect to time and space. Stereo vision is a technology that is well suited for delivering a precise description within its field of view. Stereo is purely a passive technology that primarily uses only two cameras and a processing unit to do the matching and 3D reconstruction.

However, for extracting dense and reliable 3D information from the observed scene, stereo matching algorithms are computationally intensive and require a high-end hardware resources. Integrating such an algorithm in an embedded system, which is in fact limited in resources, scale, and energy consumption, is a delicate task. The real-time requirements of most robot applications complicate the realization of such a vision system as well. The key to success in realizing a reliable embedded real-time-capable stereo vision system is the careful design of the core algorithm. The trade-off between execution time and quality of the matching must be handled with care and is a difficult task. The definition of the term real-time by Kopetz [70] which means that a task has to be finished within an a priori defined time frame is extended in this work. Additionally, demands on fast (at least 10 fps), constant, and scene-independent processing time are made.

* Corresponding author. Fax: +43 50 550 4150.
E-mail addresses: martin.humenberger@ait.ac.at (M. Humenberger), christian.zinner@ait.ac.at (C. Zinner), michael.weber@ait.ac.at (M. Weber), wilfried.kubinger@ait.ac.at (W. Kubinger), vincze@acin.tuwien.ac.at (M. Vincze).

1077-3142/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2010.03.012

In this paper the challenge of fast stereo matching suitable for embedded real-time systems is tackled. An adapted, high speed and quality stereo matching algorithm especially optimized for embedded systems is presented. Furthermore, an evaluation of the results using the Middlebury stereo evaluation website and real-world scenarios is given and experimental results of reference implementations on a Personal Computer (PC), a Digital Signal Processor (DSP) and a Graphics Processing Unit (GPU) are presented. The remainder of this paper is organized as follows: Section 2 introduces a summary of the fundamentals of stereo vision and the state-of-the-art in stereo matching algorithms. Section 3 gives a detailed description of the proposed real-time stereo engine. The algorithm's parameters are analyzed in detail in Section 4 and Section 5 shows the reference implementations on a PC, a DSP and a GPU. Finally, Section 6 presents evaluation results of our algorithm and Section 7 concludes the paper and gives an outlook to future research.

2. Stereo vision

The main challenge of stereo vision, also called stereopsis, is the reconstruction of 3D information of a scene captured from two different points of view. This can be done by finding pixel correspondences between both images. The horizontal displacement of corresponding pixels is called disparity. Classical stereo vision uses a stereo camera setup built up of two cameras, called a stereo camera head, mounted in parallel. It captures a synchronized stereo pair consisting of the left camera's and the right camera's image. A typical stereo head is shown in Fig. 1; the distance between both cameras is called the baseline.

Once the correct disparity for a pixel is found, it can be used to calculate the orthogonal distance between one camera's optical center and the projected scene point with

$$z = \frac{b \cdot f}{d} \tag{1}$$

where d is the disparity, b the baseline and f the camera's focal length. If 3D data should be given in camera coordinates, (2) can be used, where K is the camera calibration matrix, the pixel is given in homogeneous coordinates $(u \cdot z_c,\ v \cdot z_c,\ z_c)^T$ and z_c is calculated with (1).

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = K^{-1} \begin{pmatrix} u \cdot z_c \\ v \cdot z_c \\ z_c \end{pmatrix} \tag{2}$$

K and f have to be determined by camera calibration which is essential for fast stereo matching. On the one hand, camera lens distortion can be removed, and on the other hand, the images can be rectified. Rectified images fulfill the epipolar constraint, which means that corresponding pixel rows share the same v-coordinate, so the search for corresponding pixels is reduced to a search along one pixel row instead of through the whole image. For this work it is always assumed that the cameras are calibrated and the stereo image pairs are rectified. Details about camera calibration can be found in Zhang [7], Sonka et al. [2], Fusiello et al. [9] and Bradski et al. [6]. Commonly used implementations are the Caltech Calibration Toolbox, Bouguet [5], and in the OpenCV library [8].

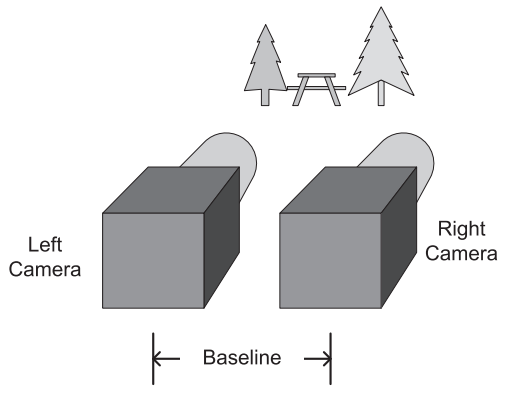

Fig. 1. Typical stereo camera head.

2.1. Related work in stereo matching algorithms

A stereo matching algorithm tries to solve the correspondence problem for projected scene points and the result is a disparity map. This is an image of the same size as the stereo pair images containing the disparity for each pixel as an intensity value. In the ideal case, each scene point visible in both images which has exactly one representing pixel per image, can be determined uniquely. In practice, this is not so easy due to the vast number of scenery points in different distances which cause a pixel in the left image for instance to be mapped to a series of similar pixels in the right image. The most common problems stereo matching algorithms have to face are occluded areas, reflections in the image, textureless areas or periodically textured areas and very thin objects. Textureless areas in particular are a major problem for stereo matching algorithms. Therefore the handling of those areas is an important aspect for the confidence of resulting matches. A good summary of many stereo matching algorithms can be found in Brown et al. [28] and Scharstein and Szeliski [29].

There are two main groups of stereo matching algorithms: feature-based and area-based algorithms. The first try to find proper features, such as corners or edges, in the images and match them afterwards, while the second try to match each pixel independently to the image content. Feature-based algorithms result in a sparse disparity map because they only get disparities for the extracted features. Area-based algorithms calculate the disparity for each pixel in the image, so the resulting disparity map can be very dense. This section gives an overview of the basic matching techniques that are currently used. The techniques introduced are restricted to area-based algorithms because this work is concerned with dense disparity maps.

Basically, an area-based stereo matching algorithm is built up as follows: First, usually pre-processing functions are applied, e.g. a noise filter. Second, the matching costs for each pixel at each disparity level in a certain range (disparity range) are calculated. The matching costs determine the probability of a correct match. The smaller the costs, the higher the probability. Afterwards, the matching costs for all disparity levels can be aggregated within a certain neighborhood window (block). In the following, popular costs calculation methods for pixel (u,v) in the reference image I_1 and the corresponding image I_2 are shown. The disparity is denoted as d and an $n \times m$ aggregation is included in the simplified notation

$$\sum_{i=n} \sum_{j=m} = \sum_{i=-\lfloor \frac{n}{2} \rfloor}^{\lfloor \frac{n}{2} \rfloor} \sum_{j=-\lfloor \frac{m}{2} \rfloor}^{\lfloor \frac{m}{2} \rfloor}. \tag{3}$$

The first method is the most popular sum of absolute differences (4), the second is the sum of squared differences (5), the third is the normalized cross correlation (6) and the last is the zero mean sum of absolute differences (7). The last two methods make the costs invariant to additive or multiplicative intensity differences caused by different shutter times, lighting conditions or apertures of the cameras.