Real-time Detection and Classification of Machine Parts with Embedded System for Industrial Robot Grasping

Hao Guo, Han Xiao, Shijun Wang, Wenhao He and Kui Yuan

Institute of Automation
Chinese Academy of Sciences
95 Zhongguancun East Road, Beijing, China
{hao.guo, han.xiao, shijun.wang, wenhao.he, kui.yuan}@ia.ac.cn

***Abstract* –In this paper, a real-time machine vision system is designed for an industrial robot to grasp from an assembly line a class of machine parts which are similar in the general shape but different in details. In order to get real-time performance, the system is implemented on an embedded image card with an FPGA (Field Programming Gate Array) accelerating the computation. The method can be divided into two stages. First, the holes and edges of the machine parts are detected from each frame with the FPGA. Then a DSP (Digital Signal Processor) chip on the image card performs the rest of the computation by identifying the location and type of each of the machine parts in the image based on the information of all the holes and edges. A rotationally adaptive edge-based template matching technique is used in our method, which not only reduces the amount of computation but also provides robustness against illumination changes. Experiments demonstate that the machine parts can be located accurately under arbitrary in-plane rotations and can be classified correctly according to the details in their shapes. Our system can run with an industrial camera at a resolution of 640×480 and a speed of 50 fps (frames per second) or higher.**

Index Terms - Machine Vision; Object Recognition; FPGA; Embedded System; Industrial Robot

I. INTRODUCTION

Highly automated manufacturing is one of the most important goals in modern industry [1][2]. Many researchers have worked in this field and various results and methods have been presented [3][4]. Since most of current robots on production lines are simply repeating a series of preset motions, the majority of past work focuses on how a robot can automatically handle a job without a vision system.

In order to enable industrial robots to perform more complex tasks, however, visual servoing systems are necessary. Unfortunately, as computer vision is a developing area far from mature, it is still a challenge to detect, locate and identify objects from images. In addition, most computer vision algorithms are very time consuming, making it even harder to develop a vision system that can provide real-time information for the control of a mechanical arm.

Nevertheless, efforts are still being made. A very popular approach to detecting and recognizing objects is to establish correspondences between visual features in the input image and those in a database image [5-8]. Arguably as the most representative one of such methods, Scale Invariant Feature Transform (SIFT) [7] detects extreme values in the scale space to get potential interest points and generates a 128-dimensional descriptor for each interest point. Histogram of Oriented Gradients (HOG) [8] is another successful algorithm of this type, which captures the local distributions of image gradients computed on a regular grid. However, the drastic appearance change of a shiny object makes it intractable to find the corresponding visual features between images of the same objects in different poses. What is more, visual feature matching only works well for objects containing rich locally textures which industrial machine parts rarely have (see Fig.1).

Another conventional way to detect and classify objects is template matching, which uses the whole template as a global feature. In this type of approach, various object images are captured and stored in a database. An object can be detected from the input image by finding a correct match in the database. C Hong employed Dominant Orientation Templates and constructed a DOT similarity map to achieve a robust performance [9]. S Hinterstoisser designed a method to handle textureless objects under small image transformations [10]. Generally speaking, this type of approach is simple and straightforward, but it has two drawbacks: long computation time and sensitivity to local appearance changes caused by occlusions, reflections, shadowings or small changes in pose.

In this paper, a hardware computation based machine vision system is designed and applied to the detection and classification of machine parts on an assembly line which requires real-time visual feedback to the robot arm. By using

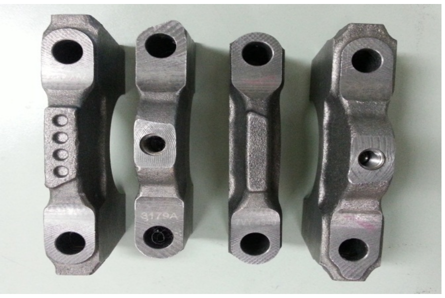

Fig. 1 Industrial machine parts

Work supported by National Natural Science Foundation of China (No. 61273360 & No. 61203328), and the National High Technology Research and Development Program of China (No. 2013AA040903).

an FPGA (Field Programmable Gate Array), all the holes in the machine parts and all the edges in the image are detected in real time from each frame. On this basis, a rotationally adaptive edge-based template matching algorithm is implemented with a DSP (Digital Signal Processor) to identify the position and type of each machine part in the image. Thanks to the parallel hardware computation in the FPGA and the robustness of our edge information based algorithm, our system can detect and classify every machine part in the input image reliably at a high processing rate.

The rest of the paper is organized as follows. Section II presents the hardware components of an industrial robot grasping system. Section III describes the algorithms implemented on the FPGA and the DSP, while Section IV explains how to optimize the method to keep a constant processing rate. In Section V, two experiments are carried out to test the feasibility and speed of the algorithm and the grasping system. Conclusions are given in Section VI.

II. Hardware Components of the System

Fig. 2 shows the overview of the system. In the following paragraphs, we introduce the hardware components of the system.

A. *ABB Industrial Robot*

The ABB 120 robot is a small sized multipurpose industrial robot which weighs 25 kg and can handle a payload of 3 kg (4 kg with vertical wrist) with a reach of 580 mm. Its accuracy or repeatability is 0.01 mm. In our experiments we use it for grasping the machine parts and transfer them to the right places according to their types. In order to get a large and safe working space, the ABB robot is hanged upside down.

B. *Intelligent Image Card*

The intelligent image acquisition and processing card (see Fig. 3) is the core of our machine vision system. The PCB (Printed Circuit Board) is designed by Dr. Wenhao He. The FPGA on the card is Altera Cyclone III EP3C40F484, while the DSP on the card is TI's TMS320DM642. In addition, there are two 512K×16bits SRAMs (Static Random Access Memory) on the card used as data buffers. The total power consumption of the FPGA and the DSP is less than 3 W.

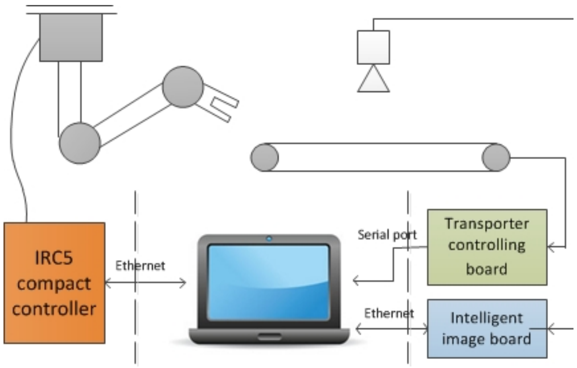

Fig. 2 Overview of the system

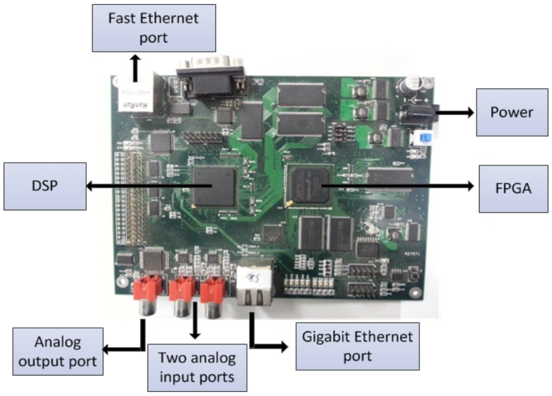

Fig. 3 The intelligent image board

The input image can either come from a digital camera through a gigabit Ethernet port or come from an analog camera through a PAL video input port.

C. *Transporter*

The transporter is placed under the camera and the robot. It is controlled by a microcontroller which is connected to a computer through a serial port. When the system begins to work, the objects (machine parts) are moved by the transporter to emulate the conveyor belt of an assembly line.

Before doing the experiments, there is a need to obtain a number of parameters, including the camera's intrinsic parameters, the transporter's speed and the spatial relationship between the camera and the robot.

III. Basic Approach

In this section, we describe our hardware oriented algorithms for machine part detection and classification. First, the offline algorithm for obtaining the edge-based templates of the machine parts is explained. After that, the online algorithm is described in details in which the FPGA undertakes the pre-processing operations while the DSP locates each machine part in the image and classifies them using the templates.

A. *Obtaining Templates*

By invoking the FPGA to get an edge image of each machine part, the DSP obtains the templates in which the width of the edges is just one pixel, as shown in Fig.4. The two circles in each template image correspond to the two holes in the machine part.

After storing a template image as well as the positions of the two circles' center points, a polar coordinate system can be established in which the axis starts from the left circle's center point and points to the right circle's center point. Every edge pixel can be expressed in this polar coordinate system. However, we have found from experiments that the edge pixels far from the origin of this polar coordinate system have large errors when expressed in polar coordinates, which will have a negative impact on the following classification procedure. To address this problem, we divide the edge pixels into three groups based on the distances from the edge pixel to

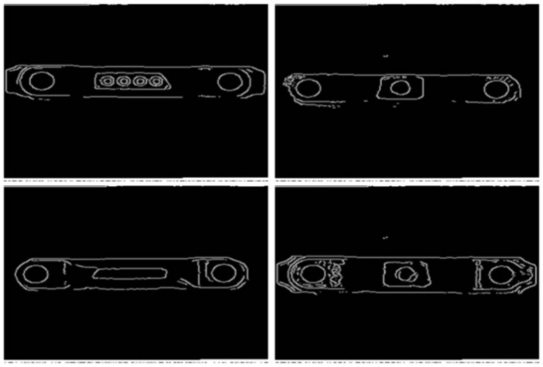

Fig. 4 Single pixel edge templates of four objects

the two circles' center points, as shown in Fig. 5. On this basis, we establish a polar coordinate for each of the three groups with the respective origins at the center of the left circle, the center of the right circle and the point in the middle. In this way, the radii of the pixels in each coordinate system are much smaller and the accuracy of the template is therefore boosted.

B. Pre-processing the Image

This step is realized in the FPGA with two parallel pipelines. The first pipeline detects edges from each frame, while the second one detects the holes in every machine part.

In the first pipeline, the FPGA applies a 5×5 Gaussian filter to the gray image to remove random noise, and then computes the gradient at each pixel's position. After that, local oriented maxima of the gradients are detected and those above a given threshold are selected as edge points. In fact, this is exactly the hardware implementation of the well-known Canny edge detector, and all the resulting edges are as thin as one pixel. In order to control the output, a register is implemented in the FPGA with its value modifiable by the DSP. According to the value of the register, the edge image can either be output to the SRAM directly or be fed into the next stage of the pipeline where a 3×3 morphological dilation is implemented. In fact, the thin edge images are used for obtaining the templates of the machine parts in the offline algorithm, while the thick edge images generated by the 3×3 dilation module are used for locating and classifying the machine parts in the online algorithm. Fig.6 shows an example of the thick edge images.

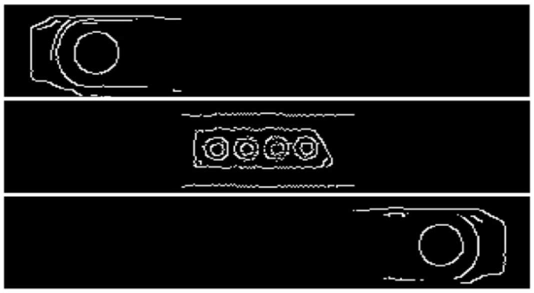

Fig. 5 Three groups of pixels in the template

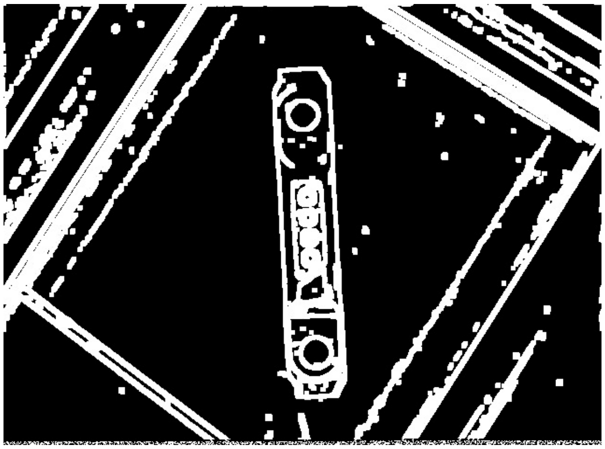

Fig. 6 A thick edge image

In the second pipeline, the original gray image is thresholded into a binary image in which a disk-shaped detector is applied with a scanning window. As shown in Fig. 7, the disk-shaped detector is a kind of binary template which is intended for the detection of the holes in the machine parts. The criterion for detection is

$$\sum_{row}\sum_{column}\delta(\text{pixel}(P)=\text{template}(P)) > circle\_threshold \qquad (1)$$

where pixel(P) represents the binary pixel value at location P, and $\delta(E)$ is a function that returns 1 if E is true, 0 otherwise. The value of this function at each pixel position is summed up in the square scanning window. If the result is larger than a given threshold, a hole is detected and the FPGA treats the center pixel of the square as the center point of the hole.

C. Merging Detection Results to Get Unique Center Points

In fact, for each hole the FPGA can get positive detection results at several positions which are not far away from one another, for we set the *circle_threshold* in equation (1) to 90% of the total area of the square. So the next step is to merge the detection results to get a unique center point for each hole. This step is performed by the DSP. First, the detected positions are divided into several groups with the distance between any two members in a group smaller than 5 pixels. After that, the average position of the members within each group is calculated and is regarded as the final center point of each hole.

D. Locating Objects

One characteristic of our machine parts is that each of them has two holes with a standard distance. To locate an object (machine part), the correct pair of holes must be found. This step is also performed by the DSP. First, all the holes

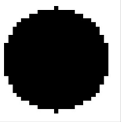

Fig. 7 The disk-shaped hole detector

detected by the FPGA are stored in a list. Then these holes are grouped into candidate pairs according to the standard distance. After that, each candidate pair is verified by checking the existence of the two parallel lines which is also a characteristic of our machine parts. The finally passed pairs of holes indicate the machine parts detected in the image. They are stored in another list and passed on to the next processing stage which classifies them according to the details in their shapes.

E. Classifying

The templates obtained with the offline algorithm (Section III-A) are used here for the classifying purpose. For each located object (machine part) in the image, the templates are used one by one to get the matching result. For a given template, the matching process is as follows.

First, the correspondences between the two holes in the located object and the two holes in the template are established. Since the object may not be symmetrical, the two possible ways of corresponding should both be tried.

On this basis, the correspondences between the edge pixels in the object and the edge pixels in the template are checked. As the pixels in each template are divided into three groups (Fig. 5), they are processed under the three respective polar coordinate systems during this process. Correspondingly, the pixels on the located object are also divided into three groups. For each group, the number of edge pixels that can find matches on the template is recorded. We represent them with S_1, S_2 and S_3. Meanwhile, for each pixel group on the object, the number of edge pixels that cannot find matches on the template is also recorded. We represent them with F_1, F_2, and F_3. Obviously, a high probability of matching between the object and the template requires that S_1, S_2 and S_3 are large while F_1, F_2, and F_3 are small. Therefore, we can define the criteria with the following inequation groups

$$\begin{cases} S_1 > S_{10} \\ F_1 < F_{10} \end{cases} \tag{2}$$

$$\begin{cases} S_2 > S_{20} \\ F_2 < F_{20} \end{cases} \tag{3}$$

$$\begin{cases} S_3 > S_{30} \\ F_3 < F_{30} \end{cases} \tag{4}$$

where S_{10}, S_{20}, S_{30}, F_{10}, F_{20} and F_{30} are six thresholds obtained with experiments. Inequation groups (2)(3)(4) are conditions corresponding to the three pixel groups on the object. As long as two of the inequation groups are satisfied, the template is considered as a candidate. This can ensure the detection rate even under bad illumination conditions while maintaining a high accuracy. In most cases, there is only one candidate template. However, if there are more than one candidate templates by chance, the one with the highest rate of matched pixels is selected as the matching result. This criterion is

$$\max\left\{\frac{\sum S_i}{\sum S_i + \sum F_i}\right\},\ i = 1,2,3 \tag{5}$$

IV. Faster Method

Actually, detecting and classifying all the objects in every frame is a waste of computation. As the objects move relatively slowly compared with the frame rate of the camera, there is no need to reclassify the objects that have already been recognized in last frame, and even the detection process can be simplified based on the results of last frame. Therefore, we optimize the method to further accelerate the processing speed.

In the new algorithm, a list containing the information of the objects in the image is maintained. Each entry in the list records the information of one object, including the object's type, the positions of the center points of the two holes, and a flag variable indicating if the positions of the holes have been updated in this frame. An example of such a list is shown in Table I.

The FPGA keeps detecting the candidate positions of every hole in each frame, and the DSP merges the candidate positions to get an unambiguous center point for each hole, as described in Section III-B and C. After that, the positions of the holes are compared with those in the list. Since the speed of our camera is 50 fps or higher, the time between two sequential frames is no more than 20 ms. In such a short time, the objects (machine parts) only move slightly on the conveyor belt of the assembly line. Therefore, most of the holes in the image can be matched to a corresponding one in the last frame with only a slight displacement. On this basis, the positions of the holes in every entry of the list are updated in each frame. If a hole in the image cannot find a match in the list, it is a new one in the image, and therefore we create a new entry in the list. On the other hand, if a hole in a certain entry of the list is not matched with a hole in the image, it means that the object has moved out of the image, and therefore we delete the entry from the list.

Using this optimized method, all the machine parts can be located and tracked with minimum computation. On this basis, the DSP can focus on one object per frame, comparing it with every template to determine its type. For example, if the image contains 10 objects, all of their types can be determined after 10 frames, while the image card can keep a constant processing rate as high as 77 frames per second.

TABLE I
An Example of the List Containing the Objects' Information

No.	Object class	Circle center point-1	Circle center point-2	Refreshed flag
…				
8	2	(86,124)	(292,125)	false
9	5	(71,253)	(277,256)	false
10	1	(142,77)	(144,283)	true
11	3	(67,140)	(272,140)	true
…				

V. Experiments and Results

In order to test the performance of our method and system, two experiments are conducted. The first experiment aims to test the feasibility and speed of the machine vision system in which the image card needs to detect and classify objects in different situations. The second experiment is designed for the industrial robot to grasp the objects according to the information provided by the embedded image card.

A. Feasibility and Speed of the Machine Vision System

Fig. 8 shows the detection and classification results. The objects appear with different rotation angles at various locations in the image. During the experiment, only a tiny number of recognitions have failed. The failed instances usually happen when the object is near the image boundary. This is because the boundary effect will result in a deformation in the shape of the object and cause a false template matching score. To resolve this problem, applying constraints on the object's location in the image is a good idea.

Table II shows the processing rate before and after optimizing the method. "Number of objects" in the table refers to how many objects appear in the image at the same time. The results demonstrate that after optimizing, the processing rate keeps at a constant value. The more objects appear in the image, the stronger the optimizing effect is. When no new object appears in the image, the processing time decreases to only 5 ms. In this case, the image card only needs to update the postitions of the holes which have moved slightly in each frame.

B. Grasping Experiment

The grasping system is shown in Fig. 9. The hardware components have been introduced in Section II. The computer controls the image card, the transporter and the robot, so there are three threads running in the control system. The first thread communicates with the image card to receive the recognition results continuously and store the information of all the objects in the computer. The second thread controls the transporter to reciprocate all the time, while the last thread reads the objects' information, transforms the objects' positions from the camera coordinate system to the robot coordinate system and controls the robot to grasp the objects one by one.

TABLE II
The Detecting Rate of the Method

Number of objects	Before optimizing		After optimizing	
	Time (ms)	Rate (f/s)	Time (ms)	Rate (f/s)
1	13	77	14	77
2	28	36	14	77
3	41	24	14	77
4	65	15	14	77

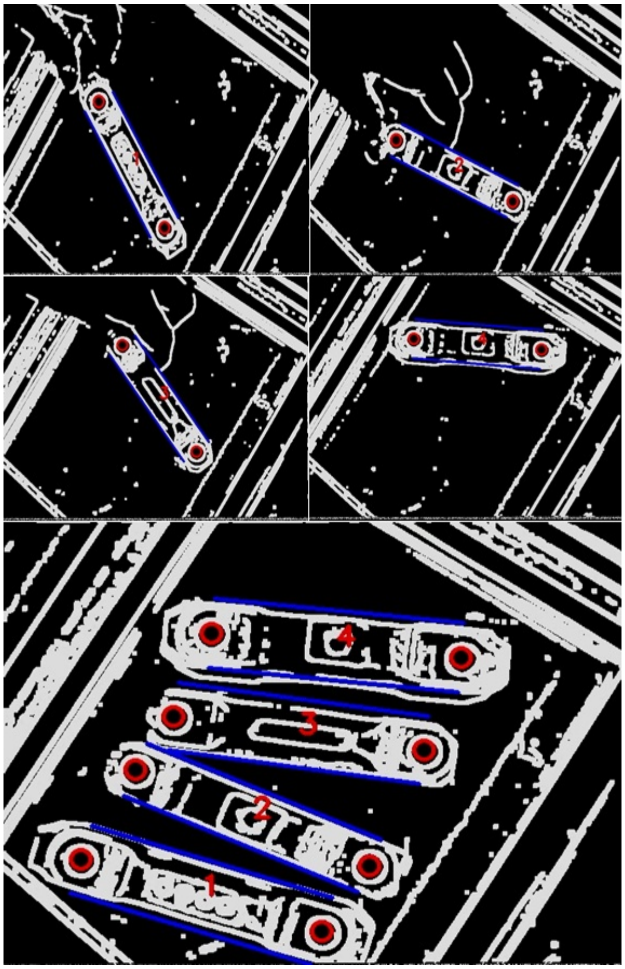

Fig. 8 Detection and classification of machine parts. The red circles indicate the locations of the holes in the machine parts while the blue lines are the verified parallel lines mentioned in Section III-D. The number in the middle of each object represents its type. (Top: four types of machine parts are detected separately; Bottom: detecting the four objects simultaneously)

VI. Conclusions

In this paper, we have presented a machine vision system which can detect and classify a class of machine parts in real time and provide the information for an industrial robot to grasp them. An FPGA plays a key role in accelerating the vision algorithm by detecting the holes in the machine parts as well as generating an edge image from each frame. A DSP, which is a kind of microprocessor, receives the intermediate results from the FPGA and implements a rotationally invariant template matching algorithm to get the final recognition results. To further accelerate the vision system, the algorithm in the DSP is optimized to reuse the detection and classification results in the next frame. Experiments have demonstrated the effectiveness of our system and methods.

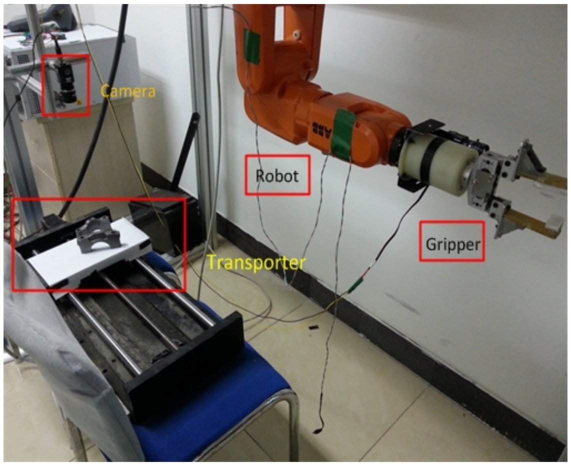

Fig. 9 The grasping system

Our future work will aim at improving the method to handle more complex cases, such as how to deal with the deformed edges when the objects rotate in a 3D space.

References

[1] R. Wilcox, S. Nikolaidis, and J. Shah, "Optimization of temporal dynamics for adaptive human-robot interaction in assembly manufacturing," 2012.

[2] D. Bortot, B. Hawe, S. Schmidt, and K. Bengler, "Industrial Robots-The new friends of an aging workforce," *Advances in ergonomics in manufacturing,* pp. 253-262, 2013.

[3] C. Canali, F. Cannella, F. Chen, G. Sofia, A. Eytan, and D. Caldwell, "An automatic assembly parts detection and grasping system for industrial manufacturing," in *Automation Science and Engineering (CASE), 2014 IEEE International Conference on*, 2014, pp. 215-220.

[4] K. Kim, S. Kang, J. Lee, and J. Kim, "Vision Based Bin Picking for Industrial Robot," 2014.

[5] P. Mainali, G. Lafruit, Q. Yang, B. Geelen, L. Van Gool, and R. Lauwereins, "Sifer: Scale-invariant feature detector with error resilience," *International Journal of Computer Vision,* vol. 104, pp. 172-197, 2013.

[6] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer vision–ECCV 2006*, ed: Springer, 2006, pp. 404-417.

[7] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision,* vol. 60, pp. 91-110, 2004.

[8] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005, pp. 886-893.

[9] C. Hong, J. Zhu, M. Song, and Y. Wang, "Realtime object matching with robust dominant orientation templates," in *Pattern Recognition (ICPR), 2012 21st International Conference on*, 2012, pp. 1152-1155.

[10] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab, "Dominant orientation templates for real-time detection of texture-less objects," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010, pp. 2257-2264.