PERGAMON

Robotics and Computer Integrated Manufacturing 18 (2002) 125–133

Robotics and Computer Integrated Manufacturing

www.elsevier.com/locate/rcim

Visual processing and classification of items on a moving conveyor: a selective perception approach ☆

H. Işil Bozma*, Hülya Yalçin

Intelligent Systems Laboratory, Department of Electrical Electronic Engineering, Boğaziçi University, Bebek 80815 İstanbul, Turkey

Abstract

Many industrial applications require some sort of automated visual processing and classification of items placed on a moving conveyor. In this paper, we present a selective perception based approach to visual processing. The novelty of this approach is that instead of processing the whole image, only areas that are deemed "interesting" and hence calling for attention are analyzed. The attentional sequences thus constructed can then be used for a variety of tasks including shape determination. Since only a small portion of the whole image is processed, visual processing can be real-time and flexible without requiring special hardware. Two different applications based on this approach are described. In a defective item detection task, we explain in detail how attentional sequences can be used. As a second application, the approach has been implemented in an automated remote controller sorter in a TV manufacturing plant—thus confirming its practical applicability. © 2002 Published by Elsevier Science Ltd.

Keywords: Visual processing; Classification; Selective perception; Attentive systems; 2D shape description; Computer vision

1. Introduction

Many industrial applications require some sort of automated visual processing and the classification of items placed on a moving conveyor [1]. A typical process comprises of (i) looking at the items on the conveyor via some type of sensor such as a camera, (ii) localizing any single item, (iii) classifying the item based on a set of features such as shape and (iv) performing the necessary action depending on the classifications made. In quality control applications, the classification output may be binary as " pass vs. defective" while in sorting tasks, the classification output may refer to the category of the particular item.

A typical setup is as shown in Fig. 1. Consider items—arbitrarily positioned and oriented—to be moving on a conveyor. A camera located above the conveyor views the items orthographically. We assume that there is an item separator placed before the camera so that the incoming items are not overlapping—which is a realistic assumption in many manufacturing environments. A sensing device signals the presence of a new item, its image is taken and an analysis is performed. Note that despite the item separator, the items may be arbitrarily positioned and there usually is some amount of perspective distortion on the image plane. The goal of the classification is to determine the shape of an item under view and whether there are any deviations from its "golden" model. For automated classification to be feasible, the following issues must be addressed effectively [2]:

- The items to be inspected may be odd-shaped—containing holes and extrusions;
- Items' shapes may not be regular;
- Items' positions and orientations may be arbitrary;
- Real-time visual processing;
- Minimal special hardware requirements and
- New items may be added frequently.

Under the strong assumptions regarding the first three issues, a variety of approaches have been proposed in the literature. However, when the items may come at arbitrary positions and be arbitrarily shaped, most of the methods turn out to be computationally too expensive and thus require special processing hardware. In this paper, we present an approach that allows the items to be arbitrarily located and be of arbitrary shape, yet does not require special hardware. The novelty of the approach is that it is based on selective

☆This research has been supported in part by Boğaziçi University Research Fund Project grants ♯AF 96A0236, and ♯AF 96HA0222 and TÜBITAK grant Misag-65, 1995.

*Corresponding author. http://www.isl.ee.boun.edu.tr.

E-mail address: bozma@boun.edu.tr (H.I. Bozma).

0736-5845/02/$ - see front matter © 2002 Published by Elsevier Science Ltd.
PII: S0736-5845(01)00035-7

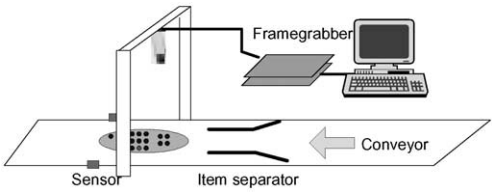


Fig. 1. Visual classification setup in an industrial setting.

perception—that is instead of processing the whole image, only areas that are deemed "interesting" and thus calling for attention are analyzed [29,30]. Since only a small part of the incoming visual data is processed, simple hardware suffices for real-time visual processing. Finally, it is simple enough so that new items may be easily added.

1.1. Related work: industrial object recognition

There have been hundreds of articles describing various methods for 2D object recognition in industrial applications [3–5]. Systems with extremely robust performance are available commercially for a wide variety of tasks including automobile, electronics and metal industries. Despite these developments, automated visual processing has been penetrating at a comparatively slow pace in many manufacturing industries [6,7]. The reasons are related to the requirements regarding (i) highly structured environments such as exact item positioning, (ii) the sufficiency of a small set of distinguishing features such as area, perimeter, etc. for identification, and (iii) specialized and usually expensive hardware [2]. This has motivated us to develop a method that may alleviate some of these problems.

1.2. Related work: selective perception

Studies in vision science have revealed that biological systems work by allocating limited computational resources to only the interesting parts of an incoming image [8–12]. This is done by saccades, rapid eye movements that direct the optical axis to a target fixation point such that the high resolution area around the fixation point—the fovea—overlaps with this interesting area [13–15,33–35]. The fovea contains almost the same number of photoreceptors as the rest of the retina and therefore can provide detailed visual information. Hence detailed processing occurs only in these regions and the features thus computed are used to solve the visual task being performed [32]. The rest of the visual field—called the periphery is much lower in resolution and serves in finding the next fixation point.

Vision researchers—motivated by these findings—have then proposed selective perception systems that mimic this type of visual processing [16–21,31,36,38–40]. Interestingly, few studies have focused on employing selective perception mechanisms in higher level tasks [37]. In this paper, we extend these ideas to automated visual processing and classification—a task assumed to be simple, yet still posing problems in real-time and robust applicability in manufacturing settings—and investigate the possibility of overcoming these problems using a selective perception based approach.

2. Selective visual processing

In selective perception systems, the processing consists of a continual repetition of pre-attention and attention stages along with cognition as shown in Fig. 2. The aim of the pre-attention stage is to determine where to look next in the visual field. After pre-attention stage, visual resources are allocated to process only a small part of the whole scene and the system goes into an attentive mode. In this mode, only this region is subjected to further processing—in order to extract more complex features. These two consecutive stages occur repeatedly—collecting data in space and time. The attentional sequences thus collected are subjected to further processing to accomplish the given visual task—eventually forming a model of the item. The cognition stage has two possible modes: (1) Learning—where the system is instructed that it is presented with a new item so that it saves this model in its memory; and (2) Classification where the system compares this model to the models in its memory and decides whether the item is close to any of those in its memory.

2.1. Fovea, periphery and visual field

Let I^t be an incoming image at time t. The fixation point at time t is the point of intersection of the optical axis with the image plane. At iteration k, a small region of I^t centered around the fixation point sized $X_f \times Y_f$ and having high acuity is designated as the fovea I_f^k. Around the fovea is a region of low resolution—referred to as the periphery I_p^k. The fovea I_f^k and the periphery together constitute the visual field I_v^k. A small region I_h^k containing fovea ensures that the same region is not looked twice. Let δI_h^k be the set of pixels on the boundary of the I_h^k. The memory M is the union of all

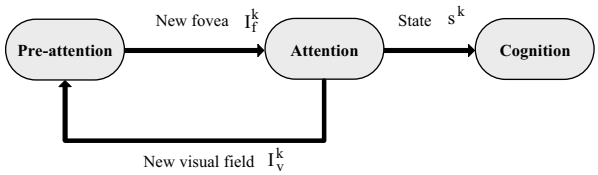


Fig. 2. General flow of processing.

inhibition regions of the foveas that have been looked at and thus enables and allows past foveas to be recalled.

2.2. Pre-attention: where to look next

The aim of pre-attention is to determine where to look next in the image I^t. Each next fovea I_f^{k+1} in a fovea sequence is found by applying simple computations on the periphery I_p^k of the current fovea I_f^k. The periphery is an adaptive sized window around the inhibition region. If no new fovea can be found, the fovea sequence ends and a new sequence begins by randomly fixating on a region not previously looked. As a result, a series of fovea sequences $I_f = (I_f^1, \ldots, I_f^{K_1}, I_f^{K_1+1}, \ldots, I_f^{K_2}, \ldots)$. The set F contains the iteration indexes of the first fovea of each fovea sequence. Finally, if all the image has been explored, the system stops. The pseudo-code for the pre-attention mechanism is

1. *Initialization*: Get the current image I^t. Initialize $M = \emptyset$. Set $F = \emptyset$. Set the iteration index $k = 0$.
2. *Finding a first fovea*: Randomly fixate on an image point in $I^t - M$ and determine the fovea I_f^k and the inhibition region I_h^k. Add index k as the index of the first fovea in this sequence via $F = F \cup \{k\}$
3. *Current periphery*: Set periphery size $n = \max(X_f, Y_f)$. Set the periphery $I_p^k = N_n(\delta I_h^k)$ where $N_n(\delta I_h^k)$ is the set of image points which are n-connected to δI_h^k.
4. *Candiate next foveas*: Let the set of candidate foveas $C(I_f^k)$ consist of all the $X_f \times Y_f$ regions in the periphery I_p^k with an $o_x \times o_y$% overlap as seen in Fig. 3. In order to avoid looking around the same fovea, the inhibition region I_h^k is excluded from consideration and thus $C(I_f^k)$ is determined from $I_p^k - I_h^k$.
5. *Saliency measure*: For each candidate fovea $I_f^c \in C(I_f^k)$ an attention criteria $a : I_f^c \to R^+$—a scalar valued function of interest based on the presence of simple features with low computational requirements—is computed. In our case, saliency is defined as the weighted sum of (i) the distance between the current fixation point and "candidate" next fixation point, (ii) the difference in their a priori selected features and (ii) the variance of these features in 8-neighborhood of the candidate next fixation point. Let us note that depending on the application, different saliency measures can easily be adopted.
6. *Next fovea*: Compute $\max_{I_f^c \in C(I_f^k)} a(I_f^c)$. If this value exceeds a prespecified threshold τ, then the next fovea is determined by
$$I_f^{k+1} = \arg \max_{I_f^c \in C(I_f^k)} a(I_f^c) \quad \text{if} \quad \max_{I_f^c \in C(I_f^k)} a(I_f^c) > \tau.$$
Compute I_h^{k+1}. In order to recall that this fovea has been previously looked at, the inhibition region is added to the memory $M = M \cup I_h^{k+1}$. Increment the iteration index $k \leftarrow k + 1$. Start loop again by going to step 3.
7. *Periphery Enlargement*: If the maximum saliency $\max_{I_f^c \in C(I_f^k)} a(I_f^c)$ does not exceed the threshold τ, increment the periphery size $n \leftarrow n + 1$. Provided that n is less than a max neighborhood bound U, go to step 4. If U is exceeded and if all the image has not been looked ($M \neq I^t$), increment the iteration index $k \leftarrow k + 1$ and go to step 2 to start a new fovea sequence. Otherwise all interesting points have been looked and the process stops.

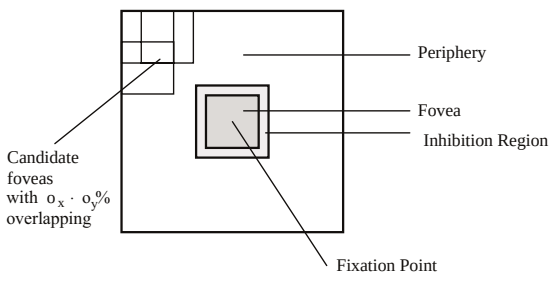


Fig. 3. Visual field and its components.

2.3. Attention

The aim of attention is to determine the state s^k of each fovea found in pre-attention. Associated with each sequence of foveas, there is a sequence of states—which we refer to as an attentional sequence. Hence, attention generates a series of attentional sequences. Attentive processing is determined by the task at hand and is much more detailed in nature than that of the pre-attentive stage. Consider N different features and let the set of values of mth feature be denoted by Ω_m. The value of each feature is obtained via an operator $f_m : I_f^t \to \Omega_m$ acting on the fovea I_f^t. If Ω_m is a finite set with N_m elements, then let $\Omega_m = \{v_{m_1}, v_{m_2}, \ldots, v_{m_{N_m}}\}$ denote the set of values that f_m can take. Let Ω denote the feature space as $\Omega \triangleq \Omega_1 \times \cdots \times \Omega_N$. Note that

$$|\Omega| = \prod_{m=1}^{N} N_m.$$

Each observation $s^k \in \Omega$ then becomes a N-vector of feature values:

$$s^k = [f_1[I_f^k], \ldots, f_N[I_f^k]].$$

The pseudo-code for the attention mechanism is

1. *Initialization*: get the current k, I_f^k and I_h^k from the pre-attentive stage.
2. *Fovea state*: apply the set of operators f_m $m = 1, \ldots, N$ and determine s^k.
3. *Adding to the attentional sequence*: add s^k to the sequence $(s^1, \ldots, s^{k-1})$.

In the results reported in this paper, we are interested in 2D shape. Thus a very simple feature set $\Omega = \Omega_1$

suffices. The set Ω_1 is defined as $\Omega_1 \triangleq \{i | i = 0, \ldots, 7\}$ where each value i indicates an edge oriented $i \times 45^\circ$. In general, more complicated features [22–24] that may be required by the task can easily be used.

3. Using attentional sequences

Each attentional sequence thus formed is input to the cognition module. In our case, since we are interested in shape-based classification, this module should generate a shape description as a first step. Recall that the items to be inspected may be odd-shaped—containing holes and extrusions. Thus, shape description should include both the items' outer shape and the shapes of all its inner parts, respectively. It turns out that since attentional sequences are based on only edge feature, each sequence in essence contains the most significant cues of a contour—of either the outer shape or an inner part—albeit perhaps partially. Thus, contour segments can be easily extracted from the attentional sequences. Furthermore, contour segments may be easily merged together to form complete and closed contours—in cases this is required. Then using a translationally and rotationally invariant transformation, each complete contour can be represented compactly by a set of few parameters. Furthermore, the geometrical relation between the outer shape and the inner parts can also be determined. Finally, depending on the modality of operation—learning or decision-making, either the generated shape description is stored in model library for future reference or compared to the stored models in order to identify the current item.

3.1. Identifying contours

Contour identification consists of two stages: (1) finding a contour segment passing through all the fixation points in an attentional sequence and (2) merging different attentional sequences—if necessary—and hence forming the maximal contour segment. The pseudo-code for contour identification is

1. *Initialization*: Get the index set F, the fovea sequence $I_f = (I_f^1, \ldots, I_f^{K_1}, I_f^{K_1+1}, \ldots, I_f^{K_2}, \ldots)$ and the attentional sequence $S = (s^1, \ldots, s^{K_1}, s^{K_1+1}, \ldots, s^{K_2}, \ldots)$ from the attention stage.
2. *Forming contour segments*:
   - 2.1. Set $i \in F$ such that $\forall j \in F$, $i \leqslant j$.
   - 2.2. Let $i_e \in F$, $i_e > i$ and $\forall k \in F$, $k \neq i$, $i_e \leqslant k$ be starting index of the next fovea sequence. If no such i_e exists, all the foveas have been considered. Skip to 3.
   - 2.3. Let $j \leftarrow i + 1$. Find a set of image points connecting the fixation point of fovea i to the the fixation point of fovea j as illustrated in Fig. 4 (left). Two image points are connected to each other if there is an 8-connectivity path from one to the other and all the image points along the path are in similar states.
   - 2.4. $i \leftarrow i + 1$. If $i < i_e$, go to step 2.3. Otherwise, set $i \leftarrow i_e$ and go to step 2.2.
3. *Merging contour segments*:
   - 3.1. Set Contours_Merged = False,
   - 3.2. For each $i \in F$
   - 3.3. For each $i_e \in F$, $i_e \neq i$. Merge the corresponding contour segments if their endpoints are located close to each other as shown in Fig. 4 (Right). Remove the index of the merged fovea sequence from F by setting $F = F - \{j\}$. Set Contours_Merged = True.
   - 3.4. If Contours_Merged = True, go to step 3.1 Otherwise stop.

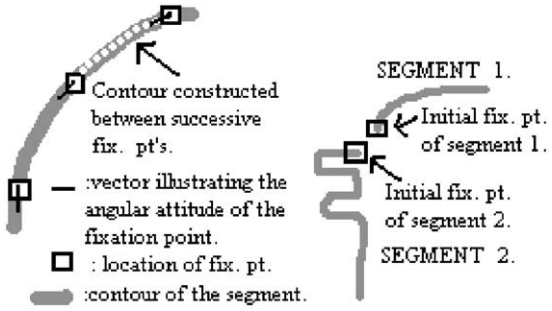


Fig. 4. Contour construction (left); merging (right).

As a result of this stage, a set of contours is generated. Each contour is a connected sequence of image points. In the next step, a translational and rotational invariant representation of each contour is generated.

3.2. Shape representation

Once contours are extracted, the next step is to represent them in a translationally and rotationally invariant manner. For this, we use elliptic Fourier descriptors [25,26]. Elliptic Fourier descriptors represent a shape weighted sum of ellipsoids. Using elliptic Fourier descriptors, each shape i is defined by a vector $q_i \in \Re^{4k+2}$ where k is the number of harmonics:

$$q_i = [a_{i0} b_{i0} a_{i1} b_{i1}, c_{i1} d_{i1} \ldots a_{ik} b_{ik}, c_{ik} d_{ik}]^T.$$

The order of harmonics k represents the accuracy of the model. Given a sequence of points forming a complete contour, a simple procedure can be used to compute the elliptic Fourier parameters [26]. A very brief description of this procedure is given in Appendix A. In order to be positionally and rotationally invariant, a set of invariants computed $p_i \in \Re^{2k}$ from the elliptic Fourier descriptors computed as described in Appendix A are used as shape parameters [28].

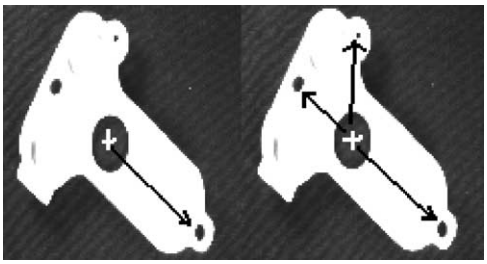


Fig. 5. Reference vector (left); Graph representation (right).

3.3. Relation between outer and inner shapes

Once the shape parameters are extracted, the next stage is to generate a model R for the shape of the item along with all of its inner parts. As an example, an item with four inner parts is shown in Fig. 5. Note that as the item may be randomly positioned and oriented, the item model must be also positionally and rotationally invariant. A graph representation is used for this purpose. The item representation is generated as follows:

1. The shape parameter p_0 of the item is added to R.
2. A subpart assumed to be always present is found and a reference vector using the center of the item and the center of an inner part is constructed as shown in Fig. 5 (left).
3. Each inner part p is added to R using the radial coordinates (r_p, θ_p) of its center point as shown in Fig. 5 (right).

3.4. Learning or identification

After the model of the incoming item is constructed, two different modes of operation is possible—depending on prior user choice. In the learning mode, the system assumes that it is presented with a new type. In this case, the item representation is stored in its memory as a "golden" model for future use—$R^m = R$. Suppose C different item types are presented to the system. Then C different models R_i^m, $i = 1, \ldots, C$ are stored in memory. Let P_L denote the number of parts of item L. Then each model consists of outer shape parameters, and P_L inner shape parameters. In the decision-making mode, a "to-be-classified" item is presented to the system and a check regarding whether the item matches a model L from the memory is made. The comparison is based on (1) comparing the outer shape parameters and (2) identifying corresponding subparts on each respectively and then (3) using a measure of proximity to determine whether the positioning and shape of each subpart on the item is as it should be. The values of all the thresholds are experimentally determined. The pseudo-code is

1. *Initialization*: Find R for the current item. Let L denote the type of current item. L is initialized to be 0 which means that type is unknown. Set $i = 1$.
2. Comparison:
   2.1. Compute $|p_{i0}^m - p_0|$—the Euclidean norm of the difference between the item's shape parameter vector and that of the ith model.
   2.2. If $|p_{i0}^m - p_0| > \lambda$, try next model by incrementing the model no $i \leftarrow i + 1$. If $i \leqslant C$, go to 2.1 else stop. All models have been tried and item cannot be identified.
   2.3. If $|p_{i0}^m - p_0| \leqslant \lambda$, $L = i$ and the current item is said to be of type L.
   2.4. For each inner part p of the model L, find an inner part $j_p \in \{1, \ldots, P_L\}$ of the item that best matches in the graph representation as

$$\mathrm{SSC}_{pj} \triangleq \sqrt{(r_p^* \cos(\theta_p) - r_{Lj}^m \cos(\theta_{Lj}))^2 + (r_p^* \sin(\theta_p) - r_{Lj}^m \sin(\theta_{Lj}))^2},$$

$$j_p \in \arg \min_{j \in \{1,\ldots,P_L\}} \mathrm{SSC}_{pj}.$$

   Here (r_p, θ_p) and $(r_{Lj}^m, \theta_{Lj}^m)$ are the radial coordinates of pth and jth inner parts in R and R_L^m respectively: If no such j_p can be found, note p as a missing part. Otherwise if the shape parameters are not similar by checking $|p_p^m - p_{jp}| > \lambda_p$, note part p as having problematic shape.
   2.5. Compute average position error ASSC:

$$\mathrm{ASSC} = \frac{1}{P_L} \sum_{p}^{P_L} \mathrm{SSC}_{pj_p}.$$

   If $\mathrm{ASSC} \leq \kappa$ where κ is a preset threshold, then perfect match. Otherwise, missing subparts.
   The ASSC value is almost invariant for a item no matter what its orientation is. Due to a missing subpart (i.e. a missing hole), the value of ASSC will change considerably compared with the model's ASSC. Having matched the subparts, we can easily determine the item matches its memory model or not.

4. Application results

This approach has been implemented in two applications—a defective item identification task and a classification task. The first development is done for a subcontracting firm producing door parts for auto industry. The second application is done for a remote controller manufacturing line in the TV industry.

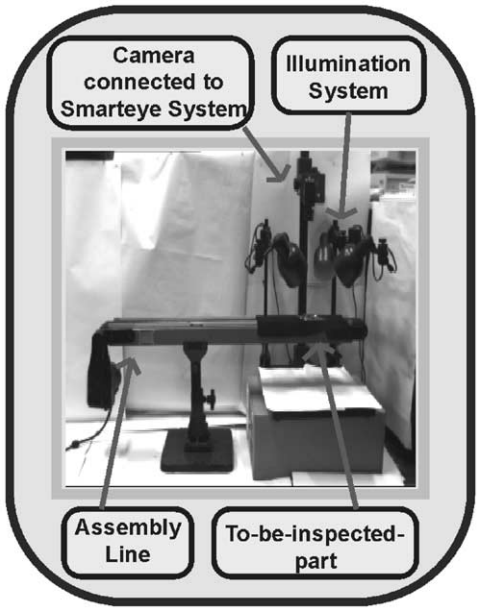
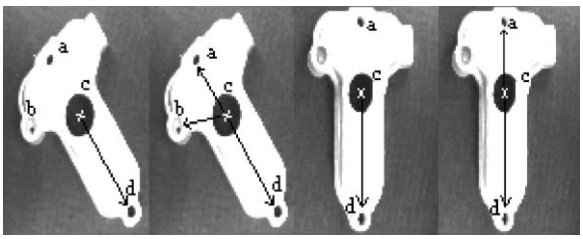


Fig. 6. Experimental setup for defective item identification.



(a) (b) (c) (d)

Fig. 7. *Item 1*—(a) and (b) *Good item*: item with inner parts and graph representation; (c) and (d) *Defective item*: defective item with missing drilled hole b and graph representation.

4.1. Defective item identification

This application is done for a subcontracting firm manufacturing metal door parts for auto industry. Items are odd-shaped—similar to that of Fig. 7. Items have several drilled holes inside them and the goal is to do 100% inspection to determine whether any has missing holes or not—and if so, to find out the missing hole. A big motivation is that due to tediousness of the task human inspectors tend to become fatigued very fast. The setup is as shown in Fig. 6. The illumination system consists of four lamps located so as to minimize the shadowing effects of each lamp. Visual processing is done on the Smarteye Vision System which is designed around a high performance DSP chip TMS320C31PQL. Computationally intensive parts of the program are directly programmed in TI assembly language.

First, we consider the item shown in Fig. 7(a) This item has four inner parts. In this item, defective items are missing a drilled hole labelled b in Fig. 7(a). If this hole has been drilled, it appears as a black small hole.

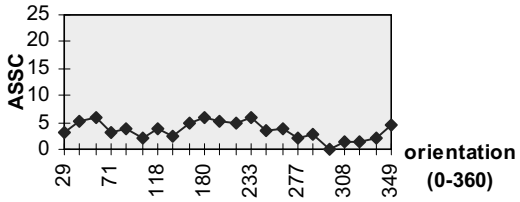


Fig. 8. *"Good" Item 1*—ASSC vs. orientation.

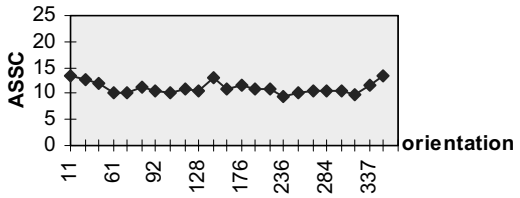
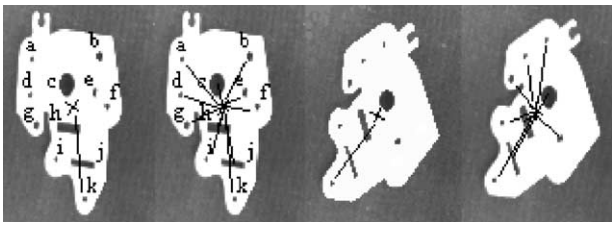


Fig. 9. *Defective Item 1*—ASSC vs. orientation.



(a) (b) (c) (d)

Fig. 10. *ITEM 2*—(a) and (b) *Good item*: item with inner parts and graph representation; (c) and (d) *Defective item*: defective item with holes b and f missing and graph representation. Note that in its graph representation arcs ending at b and f are missing.

On the other hand, if it is missing, only a dent is seen similar to that of Fig. 7(c). First the item is taught to the system and a model R_1^m is stored in memory. The inner part labelled c is used to construct the reference vector.

Then "good" items are placed with varying orientation and ASSC values are computed. The results are displayed in Fig. 8—where each point represents the mean of ten sample inspections. It is observed that ASSC varies between 0 and 6. Observe that somewhere around 300°, ASSC goes down to zero which means the model R_1^m was generated with an item oriented roughly at 300°. For the remaining orientations, ASSC is slightly different from zero since our items have extrusions and holes on them, foreshortening effects come into play and distort the image very slightly—thus causing variation of the ASSC.

ASSC values for varying orientations of the defective item are shown in Fig. 9—where again each point represents the mean of ten sample inspections. The ASSC value varies between 10 and 15. Hence a threshold value about λ, $\kappa \cong 6-7$ can be used for classifying defective items.

Similar experiments were held for another more complicated item shown in Fig. 10. This item has ten

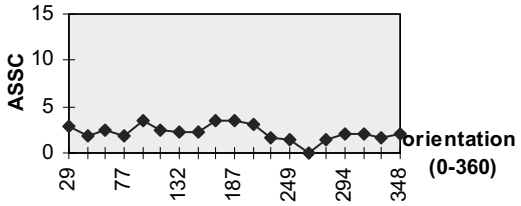


Fig. 11. *"Good" Item 2*—ASSC vs. orientation.

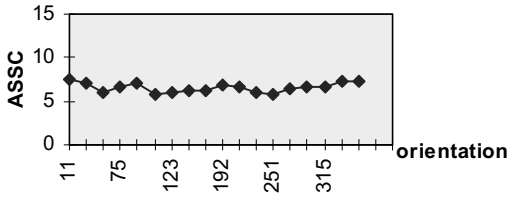


Fig. 12. *Defective Item 2*—ASSC vs. orientation .

inner holes—labelled "a"–"h" as seen in Fig. 10(a). One or several of these holes may be missing. In our experiments, defective items have holes labelled b and missing as seen in Fig. 10(c). Fig. 11 shows the mean ASSC values for 10 sample inspections for each orientation. The model of the industrial item is constructed using a "good" item with a randomly selected orientation at 270°. For the remaining orientations, again ASSC is slightly different from zero again due to foreshortening effects. Next "defective" items —where there are two holes missing—are presented to the system in varying orientations. Fig. 12 shows average ASSC values for 10 experiments. The observable difference in ASSC values between good and defective items can be used to determine a threshold λ and κ for deciding whether a given item is acceptable or not. In our case, this value is about $\lambda, \kappa \cong 4$. In addition to detecting faulty items, the system easily identifies the faulty subparts. The experiment is repeated for items with many different types of faults. As expected, as the number of faults of type "missing holes" increases, ASSC also increases.

4.2. Automated sorting

In our second application, an automated remote controller sorting system has been developed for a TV manufacturing plant—currently operational as shown in Fig. 13. There are five different types of remote controllers with about ten different colors. Each type of remote controller can be distinguished based solely on the outer shape. In this application, all the different types of remote controllers are being manufactured on the same assembly line. After being manufactured, they are subjected to functional testing—which varies according to their type. Hence, an automated visual sorter station—placed on the assembly line—ensures

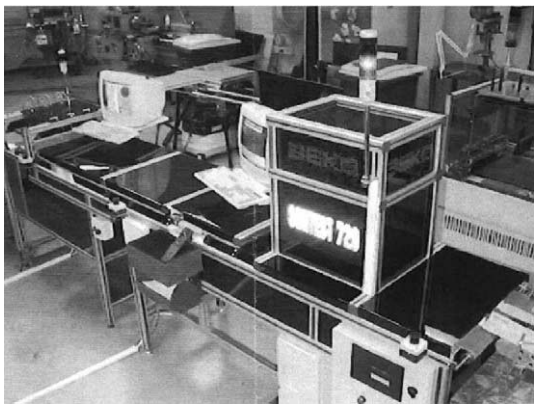


Fig. 13. Automated sorting system (courtesy of Beko A.Ş.). From the right part, remote controllers are fed automatically one-by-one from the assembly line. A camera acquires their image and visual processing determines their type. Accordingly, the remote controllers are directed to one of the control stations. Those whose types are unidentified are directed to a basket as seen in the front.

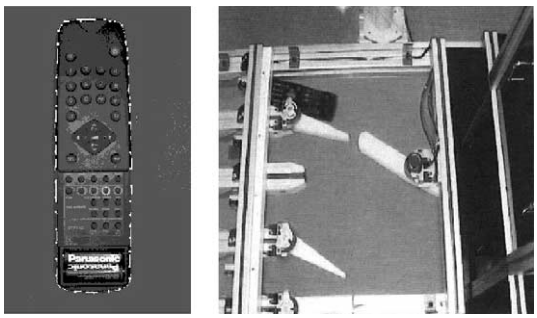


Fig. 14. Left: A remote controller and a sequence of fixation points in white dots; Right: Directing to one of the control stations.

that an incoming remote controller is sent to the appropriate control station. First, the system is presented with a sample of each remote controller type. It selectively attends to the image—thus coming up with a sequence of fixation points as shown in Fig. 14 (left). It then finds the contour segments going through these fixation points and then merges them to find the closed outer shape as outlined in Sections 2.3. It is then set to learning mode where the parametric representation p_{i0}^m of this outer shape is retained for future use. This is repeated for each type $i = 1, \ldots, 5$. After being presented with samples of all the types, the system is ready to operate in the sorting mode. When a "to-be-sorted" remote controller comes to the sorting station, it is selectively attended and the attentional sequence thus generated is used to eventually generate the invariant parameters p_0 of its outer shape. Since the outer shape is sufficient for determing its type, only steps 2.1–2.2 of the comparison step of Section 3.4 are applied. First, this parameter set is then compared with those of the models—previously taught to the system using Euclidean metric $|p_{i0}^m - p_0|$ as discussed in Section 3.4. The

type L of model that minimizes $|p_{i0}^m - p_0| \leqslant \lambda$ is designated to be type of the remote controller. According to this value, the remote controller is directed to one of the possible control stations through an electro-mechanical setup as shown in Fig. 14(right). If no such L can be found, the system cannot assign the incoming remote controller to any of the five classes and the situation is announced as a miss. In this case, the remote controller is sent to a basket. With extensive testing, false detection rates have been reduced to 0% for all types and misses to 0.5%.[1]

5. Conclusion

Within the general problem of automated classification on a moving conveyor, this work proposes a novel approach to visual processing based on selective perception. Here instead of processing the whole incoming camera image, only "interesting" regions are processed. Thus the required computational resources for being flexible and real-time at the same time are reduced considerably. In this approach, the visual processing consists of a continuum of pre-attentive and attentive stages. An attentional sequence thus generated represents the visual data spatio-temporally. This processing is occasionally followed by cognition, where attentional sequences are processed according to the demands of the task at hand. Two applications based on this approach are described. Both applications require real-time decision making. In an defective item detection task, attentional sequences are used to construct a representation of an incoming item. The average similarity measure ASSC between the model previously stored and the "to-be-inspected" item is computed in order to distinguish between good and defective items. ASSC values lying above a threshold are labelled as being defective. In the second application, an implementation of this approach for the automated sorting of remote controllers in a TV manufacturing plant is described. Here, with an extensive development and experimental evaluation, nearly 100% reliability is achieved. Furthermore, selective visual processing allows inspection times of about 100 ms. As future work, we plan to work on extending this framework to incorporate more complex features in order to be able to detect other types of defects.

Acknowledgements

This research has been supported in part by Boğaziçi University Research Fund grant # 97HA201 and TÜBİTAK grant Misag-65, 1995. The project on the automated sorting of remote controllers has been made possible with cooperation of BEKO Elektronik A.Ş. İstanbul. S. Burak Göktürk has been a major contributor to the project. We also gratefully acknowledge the contributions of Ümit Baştuğ, Metin Özdilek and the rest of the BEKO team to the mechanical design and implementation efforts. The authors also thank the anonymous readers for their suggestions regarding the improvement of the manuscript.

[1] Due to the commercial nature of the application, we cannot provide any further data.

Appendix A. Computational details for elliptic Fourier descriptors

In this Appendix, we describe a simple procedure described in detail [26] in order to compute the elliptic Fourier parameters for ith object. Each shape i is defined by a vector $q_i \in P \subseteq \Re^{4k+2}$ using EFDs, where k is the number of harmonics:

$$q_i = [a_{i0} b_{i0} a_{il} b_{il}, c_{il} d_{il} \ldots a_{ik} b_{ik}, c_{ik} d_{ik}]^{\mathrm{T}}.$$

The EFDs are computed as follows:

$$a_{ik} = (1/(\pi \omega_i k^2)) \sum_{p=1}^{P} (\Delta x_{ip}/\Delta t_{ip})(\cos(k\omega_i t_{ip}) - \cos(k\omega_i t_{i,p-1})),$$

$$b_{ik} = (1/(\pi \omega_i k^2)) \sum_{p=1}^{P} (\Delta x_{ip}/\Delta t_{ip})(\sin(k\omega_i t_{ip}) - \sin(k\omega_i t_{i,p-1})),$$

$$c_{ik} = (1/(\pi \omega_i k^2)) \sum_{p=1}^{P} (\Delta y_{ip}/\Delta t_{ip})(\cos(k\omega_i t_{ip}) - \cos(k\omega_i t_{i,p-1})),$$

$$d_{ik} = (1/(\pi \omega_i k^2)) \sum_{p=1}^{P} (\Delta y_{ip}/\Delta t_{ip})(\sin(k\omega_i t_{ip}) - \sin(k\omega_i t_{i,p-1})).$$

Δx_{ip} and Δy_{ip} are incremental changes of the ith contour in x and y directions during the time Δt_{ip}. T_i is the period to trace the ith contour once. ω_i is defined as $2\pi/T_i$.

The Euclidean invariants can be defined as follows:

$$I_{ik} = a_{ik}^2 + b_{ik}^2 + c_{ik}^2 + d_{ik}^2,$$

$$J_{ik} = |a_{ik} d_{ik} - b_{ik} c_{ik}|.$$

I_{ik} being the square sum of two semi-axis lengths of the kth ellipse and J_{ik} being the area of kth ellipse.

References

[1] Batchelor BG, Braggins DW. Commercial vision systems. In: Torras, editor. Computer vision: theory and industrial applications. New York: Springer, 1992. p. 405–52.

[2] Mattone R, Campagiorni G, Galati F. Sorting of items on a moving conveyor belt. Part 1: a technique for detecting and classifying objects. Robotics Comput Integrated Manuf 2000;16:78–80.

[3] Ballard DH, Brown C. Computer vision. Englewood Cliffs, NJ: Prentice-Hall, 1982.
[4] Haralick R, Shapiro L. Computer vision. Englewood Cliffs, NJ: Prentice-Hall, 1985.
[5] Chellappa R, Sawchuk A. Digital image processing and analysis. IEEE Catalog No. EHO232-9, 1985.
[6] Newman TS, Jain A. A survey of automated visual inspection. Comput Vision Image Understanding 1995;61(2):231–62.
[7] Kennedy CW, Hoffman EG, Bond SD. Inspection and gaging. New York: Industrial Press, 1987.
[8] Gouras P. Oculomotor System. In: Schwartz JH, Kandel ER, editors. Principles of neural science. Amsterdam: Elsevier, 1988.
[9] Kelly JP. Anatomy of central visual pathways. In: Schwartz JH, Kandel ER, editors. Principles of neural science. Amsterdam: Elsevier, 1988.
[10] Noton D, Stark L. Eye movements and visual perception. Sci Am 1971;224(6):34–43.
[11] Zeki S. The visual image in mind and brain. Sci Am 1992;267(3):69–76.
[12] Hubel DH. Eye, brain and vision. ScientificAmericanLibrary, 1988.
[13] Henderson, M.Visual attention and the attention-action interface. In: Akins K, editor. Perception. Oxford: Oxford University Press, 1996. p. 290–316.
[14] Stark L, Ellis SR. Scanpaths revisited: cognitive models direct active looking. In: Fisher, Monty, Senders, editors. Eye movements: cognition and visual perception. Erlbaum, NJ, 1981. p. 193–226.
[15] Julesz B. Dialogues on perception. Cambridge, MA: MIT Book Press, 1995.
[16] Ballard DH, Brown CM. Principles of animate vision. CVIP: Image Understanding, 1992;56(1):3–21.
[17] Ballard DH. Animate vision. Artif Intell 1991;48:57–86.
[18] Barrow H, Tenenbaum JM. Computational vision. Proc IEEE 1981;69:572–5.
[19] Rimey RD, Brown CM. Selective attention as sequential behavior: Modeling eye movements with an augmented hidden markov model. Technical Report, University of Rochester, Computer Science Department, February 1990.
[20] Swain MJ, Stricker MA. Promising directions in active vision. Int J Comput Vision 1993;11(2):1090126.
[21] Carl Fredrik Westin. Attention control for robot vision. Proceedings of the CVPR'96, 1996. p. 726.
[22] Gallant JL, Van Essen DC, Nothdurft HC. Two-dimensional and three dimensional texture processing in visual cortex of the Macaque Monkey. In: Papathomas TV, Chubb C, Gorea A, Kowler E, editors. Early vision and beyond. Cambridge, MA: MIT Press, 1995. p. 89–98.
[23] Sagi D. The psychophysics of texture segmentation. In: Papathomas TV, Chubb C, Gorea A, Kowler E, editors. Early vision and beyond. Cambridge, MA: MIT Press, 1995. p. 69–77.
[24] Nakayama K, He ZJ. Attention to surfaces: beyond a cartesian understanding of focal attention. In: Papathomas TV, Chubb C, Gorea A, Kowler E, editors. Early vision and beyond. Cambridge, MA: MIT Press, 1995. p. 69–77.
[25] Lin C, Hwang C. New forms of shape invariants from elliptic Fourier descriptors. Pattern Recognition 1987;20(5):535–45.
[26] Frank PK. Elliptic Fourier features of a closed contour. Comput Graphics Image Process 1982;18:236–58.
[27] Kunl FP, Giardina CR. Elliptic Fourier features of a closed contour. Comput Graphics Image Process 1982;18:236–58.
[28] Soyer Ç, Bozma, HI. Further experiments in classification of attentional sequences: combining instantaneous and temporal evidence. Proceedings of the ICAR'97, California, USA, 1997. p. 991–6.
[29] Soyer Ç, Bozma HI, İstefanopulos Y. A mobile robot with a biologically motivated vision system. Proceedings of the IROS'96, Japan, 1996. p. 680–7.
[30] Abbott AL. A survey of selective fixation control for machine vision. IEEE Control Systems 1992. p. 25–31.
[31] Treisman A, Gelade G. A feature integration theory of attention. Cog. Psychol. 1980;12:97–136.
[32] Kapoula Z, Robinson DA. Saccadic undershoot is not inevitable: saccades can be accurate. Vision Res 1986;26(5):735–43.
[33] Malinov IV, Epelboim J, Herst AN, Steinman RM. Characteristics of saccades and vergence in two kinds of sequential looking tasks. Vision Res 2000;40:2083–90.
[34] Clark J. Spatial attention and latencies in saccadic eye movements. Vision Res 1999;39:585–602.
[35] Ballard DH. Animate vision. Artif Intell 1991;48:57–86.
[36] Ullman S. High-level vision. Cambridge, MA: MIT Press, 1996.
[37] Koch C, Ullman S. Selecting one among the many: a simple network implementing shifts in selective visual attention. A.I. Memo 770, C.B.I.P Paper 003, 1994.
[38] Itti L, Koch C. A model of saliency-based visual attention for rapid scene analysis. IEEE Trans Pattern Anal Machine Intell 1998;20(11):1254–9.
[39] Itti L, Koch C. A comparison of feature based combination strategies for saliency-based visual attention systems. SPIE Human Vision and Electronic Imaging IV, vol. 3644, San Jose, CA, 1999. p. 373–82.