Selective Attention by Integration of Vision and Audition

T. Lourens[1], K. Nakadai[1], H. G. Okuno[1], and H. Kitano[1]

Kitano Symbiotic Systems Project
ERATO, Japan Science and Technology Corporation
Mansion 31 Suite 6A, 6-31-15 Jingumae, Shibuya-ku, Tokyo 150-0001, Japan
{tino, nakadai, okuno, kitano}@symbio.jst.go.jp

Abstract. Selective attention is one of the tasks humans solve with great ease, still in computer simulations of human cognition this is a very complicated problem. In humanoid research it even becomes more complicated due to physical restrictions of hardware. Compared to a human, camera's, e.g., have small visual fields and low resolution while motion causes a lot of noise, which makes audition a more complicated task. Combining vision and audition in humanoids is beneficial for both cues: vision because it does not suffer from noise, while audition is not restricted to an approximately $40^\circ \times 40^\circ$ receptive field area, neither to partly or fully occluded objects. Low localization accuracy of both human ($\pm 8^\circ$) and artificial ($\pm 10^\circ$) audition systems can be compensated for by using vision.

In this paper we propose a model that simulates selective attention by integrating vision and audition. A learning mechanism is incorporated as well to make the model adaptive to any arbitrary scene. The input of the model is formed by specific and robust features that are extracted from a huge amount of sensor data, hence part of the paper will focus on feature extraction.

Audition is employed to improve selective attention because objects can be occluded or outside the visual field of a camera or human vision. Visual fields can be made wider by lenses, but never reach the full 360 degrees, hence a map is needed. This map contains information about all recognized objects over time, where objects are represented by features in a symbolic description. This map, in fact, represents a kind of artificial (temporal) memory. The location information of the objects (given by real world coordinates) is stored in the map as well. Also features from both vision and audition cues are integrated in this map. Storing information over time in such a map facilitates and speeds up the selective attention model. The map can be easily extended to incorporate extracted features from other types of sensors. In a simple natural environment the functionality of the model as well as the symbiosis between vision and audition are illustrated. The scenario will show that interaction between vision and audition is beneficial which is found rarely in literature. Promising results of the scenario show that audition was needed to localize an initially invisible object, while vision after that was used to accurately localize the object.

1 Introduction

The goal of the research reported in this paper is twofold: one to model active perception in a dynamic environment by selective attention; two by multi-modal integration of vision and audition cues to improve perception capabilities. The performance of the model for localization of an object (needed for attention) as well as the need of interaction between vision and audition are illustrated by an experiment. This experiment carried out in a real world environment is kept relatively simple for two reasons: one to illustrate the need of vision and audition, and two because certain parts in the complicated chain of data processing need to be improved considerably to allow more complex scenarios. An upper-torso humanoid robot (Fig. 1) is used as a platform of this research.

Selective attention is a very complicated problem because many different aspects should be considered and integrated. For example, attention can be driven by vision, audition, and tactile sensing, or a combination of them. Attention is often (partly) driven by knowledge. Even in daily life, we often cannot say why we focus on a specific target, and the answer is: "it just happened".

Human beings have a remarkable ability to interpret a complex environment very quickly. To process this huge amount of data, a subset of data is selected. Most likely to reduce the complexity of the scene in a spatial way, a process known as *focus of attention* [33,39]. Most of the current models based on visual attention use the *feature integration theory*, explaining human visual search strategies [19,38]. Initially all features feed a map in a purely feedforward manner which topographically codes local conspicuity over the entire scene. In primates, this map is the posterior parietal cortex [9] as well as various maps in the pulvinar nuclei of the thalamus [34].

There are many ways to solve the selective attention problem. For example Itti et al. [15] used a feedforward architecture that feeds visual features into a salience map. We will propose one that is driven by both audition and vision cues, which is found rarely in literature. In active vision and audition huge amounts of data need to

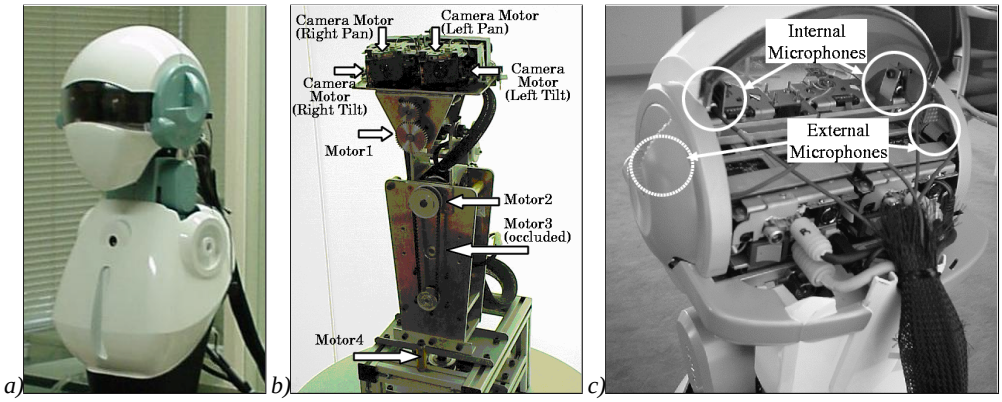


Fig. 1. Humanoid. *a)* Cover. *b)* Mechanical structure. *c)* Internal microphones (top) and cameras.

be processed, hence one of the challenging parts is to keep relevant data only. We use a very simple event driven mechanism by calculating the differences at two different time stamps to eliminate highly redundant information. An event is generated if data differs sufficiently. *Feature extraction* mechanisms should further reduce the amount of data. These features, which form the input for our model, should be very specific, since they will be stored in a map. This map can therefore be interpreted as artificial memory. This implies that the performance of the model strongly relies on the features, they need to be general, robust, and specific. Obtaining such features is difficult and time-consuming, thus another important reason to use a map is to save computational time.

Vision contains a wide variety of features, commonly the features are divided into four different groups: form, color and texture, motion, and stereo. From all those cues, features are extracted to get a robust vision system. Data obtained by vision and audition, in common, is highly redundant, but there are cases where vision and audition benefit from each other. The focus of this paper will be mainly on these cases. The visual field of a camera is usually about $45° \times 34°$. This is rather small for selective attention. Audition is used to overcome this limitation. On the other hand audition has a coarse resolution which is compensated for by using the accuracy of vision [30].

In this paper a model for selective attention by creating a map of the local environment containing symbolic information will be presented. We will focus on cases where vision and audition benefit from each other. A simple model for selective attention is presented that include a learning mechanism; Sect. 2. The input of the model consists of features that are extracted from vision and audition cues.

All aspects of used features are mentioned in Sect. 3. Corner and edge enhancement will be discussed in Sect. 4. Section 5 will use these corner and edge enhanced data for corner and edge (contour) extraction to construct a graph. Section 6 will describe how visual objects can be recognized by graph matching. In Sect. 7 we elaborate on the technical details of feature extraction in audition.

In Sect. 8 the hardware setup and a scenario will be given to illustrate how different aspects of vision and audition can benefit from each other. The scenario is kept relatively simple, since we want to stress on the symbiosis between vision and audition, rather than to create highly complex scenes. We also will assume that learning already took place. Not much attention will be paid to learning, although the model incorporates a learning mechanism. Focus of attention is necessary in active vision and audition, not only to reduce the huge amount of data, but to simulate intelligent behavior of a humanoid robot, instead of passively processing all data. We created an algorithm for selective attention based on changes in the map which happen over time. Strong attention is paid to newly appearing objects in the scene, while moderate attention is given to moving objects in the map or scene. Results of the model by playing a simple scenario are in Sect. 9. The last sections give observations, conclusions, and future research.

2 A Model for Selective Attention

In Fig. 2, a computational model of selective attention with a learning mechanism is illustrated. The first step in this model is to *extract relevant features* from the input data. By relevant features, we mean that preferably no knowledge is necessary to extract features, and thus they are generally applicable.

From different types of sensors, features will be extracted separately. All these features form continuous input in temporal domain to the whole attention model. Features from different sensors, as well as from the attention model itself, are combined in a way that ambiguities and inconsistencies are removed.

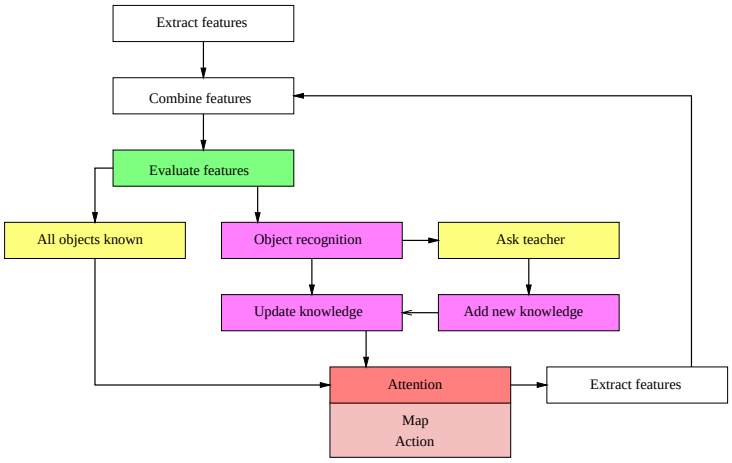


Fig. 2. Scheme for selective attention with a learning mechanism.

After evaluation of features, module “Evaluate features” in Fig. 2, one might obtain a situation where all objects are known. This is possible because only newly obtained information is added to the map; see module “Map” in Fig. 2. The map in its turn returns all features stored in the current map to inhibit the continuous input feature stream. This implies that, if nothing changes in the scene, no action takes place. Hence, attention is a process driven by changes in the map.

If features are available after feature evaluation, i.e. there are one or more new (or moving) objects in the scene, they will be forwarded to the recognition module. This module will find all known objects in the scene by performing an inexact graph matching algorithm for visible objects and currently a simple frequency separation mechanism for sound. In case not all features are evaluated it will ask the operator what a set (cluster) of features means. This is learning by supervision in a similar way as a child perceives information, processes the data, and asks a parent. The advantage of such a learning mechanism is that one obtains a compact database that is adaptive to any input. In module “Update knowledge” the newly obtained knowledge from the supervisor and recognized objects are fused and forwarded to the attention module. In the attention module all data is gathered and decisions about updating the map, focus of attention (and thus robot motion) are made. Finally, the module forwards all data to the feature extraction module, where all object features are extracted and used in turn to inhibit the feature stream.

The knowledge database is a combination of a set of features with relations between features. The map is a combination of spatial and temporal relationships with objects in the knowledge database.

3 Feature Extraction

In the current model three cues are used for vision: form, stereo, and motion. Color is used only in case of ambiguities. For audition two cues are used: pitch and direction. In this section we describe all used features briefly.

3.1 Form

Two form cues are used: *edges* and *corners*. They form the cornerstones of both recognition and stereo matching.

An edge enhancement algorithm based on a model for complex cells in the visual cortex is used [11,28]. Generalizing the algorithm to color channels in a biologically plausible way and using a multiscale approach yields accuracy to both lines and edges, and robustness to high frequency texture, noise, varying contrast, and ragged edges. The complex cells constitute input to end-stopped cells, they enhance corners and line ends. The algorithm for corner detection yields good and accurate results in the presence of high frequency texture, noise, varying contrast, and rounded corners. Edge and corner enhancement and extraction will be elaborated on in Sect. 4 and 5, respectively.

3.2 Stereo Vision

From stereo vision, an additional cue can be extracted: *depth*. Stereo is very useful on a short distance for hand-eye coordination. Although human vision uses stereo, it does not use a real world coordinate calibration mechanism.

If a person is asked to evaluate the size of a room with no reference points, like size of a door, results will be inaccurate. Most probably the brain processes features first and uses a matching mechanism to perceive a three dimensional scene.

In the model, no smart stereo matching mechanism is used, yet. To avoid complicated camera calibration techniques, a special arrangement of the stereo camera rig, called the *canonical configuration*, is used. The baseline is aligned to the horizontal coordinate axis, the optical axes of the camera are parallel, the epipoles move to infinity, and the epipolar lines in the image planes are parallel. This configuration is simple and the stereo correspondence is to be found by a human operator who will find matching points linewise to be easier, see also Sonka et al. [36]. Some authors like Mohr [26] report practical problems with the canonical stereo configuration, which adds unnecessary technical constraints to the vision hardware. Especially when high accuracy is an issue.

In Sect. 6.5, we propose a stereo matching method that is based on graph matching (Sect. 6). The algorithm contains two stages. Objects in left and right image will be recognized first. Stereo correspondences are found by matching the recognized objects obtained at the first stage of left and right images. A similar approach based on matching of edges used by Ueshiba et al. [40] might be faster, but it is difficult to evaluate the quality of this algorithm.

3.3 Motion

Audition and selective attention are useless if there is no temporal space available. For vision, this implies that motion is included as well.

Up to 30 frames (images) per second can be obtained from the humanoid. However in the current model, the processing time is far larger than real time, hence an active system with events marked by time stamps is used. When an event occurs a single frame is processed.

In the model, motion features are: *speed* and *direction* of an object. By recognizing an object and comparing it with the previous position, direction and speed can be derived easily. These parameters are especially useful for object tracking by the humanoid.

3.4 Sound

In humanoid robots, noise cancellation is a problem we cannot avoid in sound processing because robots inevitably make noise. We use a cover to make it easier to cancel noises. The cover can separate inner world of the robot from the outer world. This means that mechanical noises from the inside of the robot would be distinguished from sounds originating from outside the robot [29].

After noise cancellation, two sound features are extracted: *pitch* and *direction*. We propose a pitch extraction method that is based upon the idea of the Binaural Harmonic-Based Stream Separation system called “Bi-HBSS” developed by Nakatani et al. [32], but much faster and more accurate than Bi-HBSS.

Sound source localization is based on auditory epipolar geometry, and is determined by calculating the interaural phase (or time) difference (IPD/ITD) between left and right channels for each subband, where every subband contains extracted harmonics, since ITD (using a correlation function) is not sufficient. The method is available and robust because it does not depend on the environment unlike the head related transfer function (HRTF) which is changeable against the environment.

In Sect. 7, noise cancellation, pitch extraction and sound source localization will be explained in more detail.

4 Complex and Endstopped Cells

Hubel and Wiesel [12,13,14] explored various visual cortical regions with micro-electrodes and divided the recorded cells into four (center-surround, simple, complex, and end-stopped) distinct classes based on a so-called *building block structure*.

The basis of this structure are formed by center-surround type of receptive fields which are found in the eye, LGN, and cortex. A group of center-surround cells which are aligned in a specific orientation give excitatory input to the following class; the *simple cells* which are found in the cortex.

Simple cells in their turn feed the complex cells. The latter constitute input for the end-stopped cells. The model proposed below skips the center-surround class, but the other classes are constructed using this building block principle.

Although the exact functional, chemical, and physical behavior of groups of cells, as well as the use of Gabor functions to model receptive field profiles of simple cells are under constant debate, we modeled the main feedforward stream in early vision for two reasons:

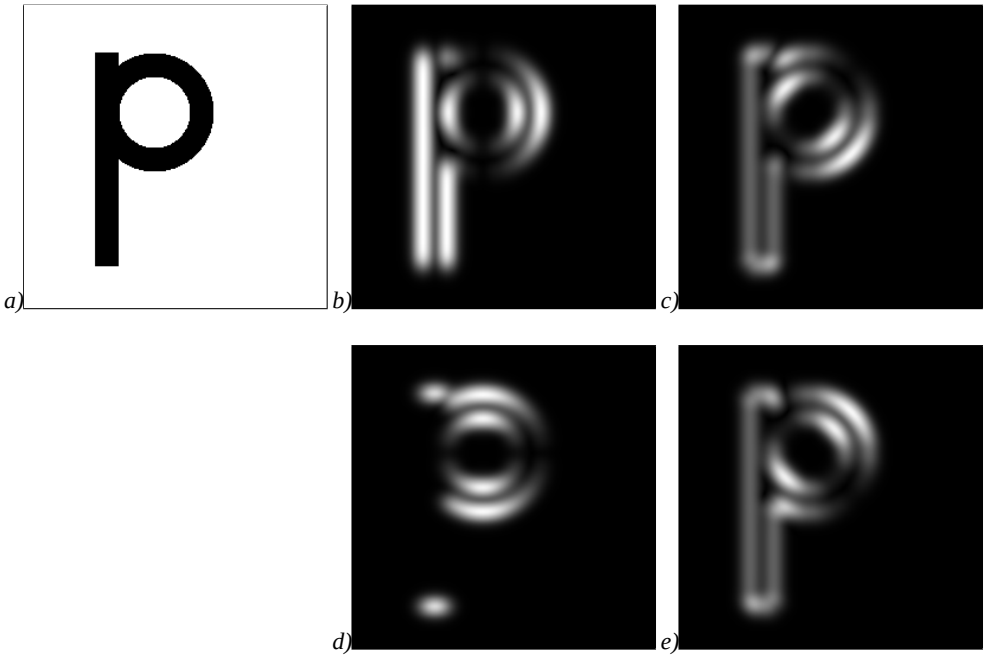


Fig. 3. Enhanced edges of *a)* P input image. *b)-e)* $\mathcal{C}_{\sigma,\theta}$ response images for $\sigma = 7.07$ and θ = 0, 45, 90, and 135 degrees, respectively.

1. to get better insights in brain theory and models
2. humans outperform any artificial vision system; by creating image processing models based on early vision we hope to get results that outperform the existing ones

4.1 Edge Enhancement

The receptive field profiles of simple cells can be modeled by complex-valued Gabor functions:

$$\hat{G}_{\sigma,\theta}(x,y) = \exp\left(i\frac{\pi}{\sqrt{2}\sigma}(x\cos\theta + y\sin\theta)\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \quad , \tag{1}$$

where σ and θ represent scale and orientation, respectively. These Gabor functions have been modified such that their integral vanishes and their one-norm (the integral over the absolute value) becomes independent of σ, resulting in $G_{\sigma,\theta}(x,y)$. They provide a transform of the image $I(x,y)$ via spatial convolution. Afterwards, only the amplitudes of the complex values are retained for further processing:

$$\mathcal{C}_{\sigma,\theta}(x,y) = ||I * G_{\sigma,\theta}|| \quad . \tag{2}$$

This "local energy representation" is the basis of all subsequent processing. A high value at a certain combination of (x,y) and θ represents evidence for a contour element in the direction orthogonal to θ. Orientations and scales are sampled linearly: $\theta_i = \frac{i\cdot 180}{N}, i = 0\ldots N-1$, $\sigma_j = \sigma_0 + j\Delta\sigma, j = 0\ldots S-1$. Figure 3 illustrates results of the $\mathcal{C}_{\sigma,\theta}$-operator, for $N = 4$ and $S = 1$.

4.2 Corner Enhancement

Starting from the local energy representation, we have developed a biologically motivated method for corner detection [45], which is described here only briefly. Our method for detecting corners yields position, sharpness, size and color and contrast. It is based on a model of cortical end-stopped cells [11].

The first step towards an end-stopped operator is an approximation of the first derivative of the $\mathcal{C}$-operator in the direction orthogonal to that of the line segment in question:

$$\hat{\mathcal{E}}^s_{\sigma,\theta}(x,y) = \mathcal{C}_{\sigma,\theta}(x + d\sigma\sin\theta, y - d\sigma\cos\theta) - \mathcal{C}_{\sigma,\theta}(x - d\sigma\sin\theta, y + d\sigma\cos\theta) \quad , \tag{3}$$

and second derivative:

$$\hat{\mathcal{E}}^d_{\sigma,\theta}(x,y) = \mathcal{C}_{\sigma,\theta}(x,y) - 0.5\mathcal{C}_{\sigma,\theta}(x + 2d\sigma\sin\theta, y - 2d\sigma\cos\theta) - 0.5\mathcal{C}_{\sigma,\theta}(x - 2d\sigma\sin\theta, y + 2d\sigma\cos\theta) \ . \quad (4)$$

These two operators are both inhibited by a tangential and a radial inhibiting operator:

$$\mathcal{I}^t_\sigma(x,y) = \sum_{i=0}^{2N-1} \left[-w_t\mathcal{C}_{\sigma,\theta_{i \bmod N}}(x,y) + \mathcal{C}_{\sigma,\theta_{i \bmod N}}(x_1,y_1)\right]^{\geq 0} \quad (5)$$

and

$$\mathcal{I}^r_\sigma(x,y) = \sum_{i=0}^{2N-1} \left[\mathcal{C}_{\sigma,\theta_{i \bmod N}}(x,y) - w_r\mathcal{C}_{\sigma,\theta_{\left(i+\frac{N}{2}\right) \bmod N}}(x,y)\right]^{\geq 0} \ , \quad (6)$$

where $x_1 = x + d\sigma\cos\theta_i$, $y_1 = y + d\sigma\sin\theta_i$, $[z]^{\geq 0}$ is equal to 0 for negative z and equal to z elsewhere (half-wave rectification), and the constants have the values $w_t = 1$ and $w_r = 4$. The corner operators on a single scale in a single direction then are:

$$\mathcal{E}_{\sigma,\theta} = \left[\left[\hat{\mathcal{E}}_{\sigma,\theta}\right]^{\geq 0} - g(\mathcal{I}^t_\sigma + \mathcal{I}^r_\sigma)\right]^{\geq 0} \ . \quad (7)$$

Constant $g = 2$ is a gain factor and for $\hat{\mathcal{E}}$ one can substitute $\hat{\mathcal{E}}^s$ or $\hat{\mathcal{E}}^d$. For details and motivation of the constants, see Würtz and Lourens [45,22].

4.3 Combining Orientations

A winner take all orientation competition mechanism (modeled by a max-operator over orientations) is used to combine different orientations

$$\mathcal{C}_\sigma = \max_{i=0}^{N-1}(\mathcal{C}_{\sigma,\theta_i}) \ . \quad (8)$$

In a similar way, at each point we consider only the maximum over all orientations, and also the maximum of single and double end-stopped operators:

$$\mathcal{E}_\sigma = \max_{i=0}^{2N-1}(\max(\mathcal{E}^s_{\sigma,\theta_i}, \mathcal{E}^d_{\sigma,\theta_i})) \ . \quad (9)$$

4.4 Combining Color Channels

With a slight and biologically justified extension of the concept, complex and endstopped cells can be extended to color channels with red-green and blue-yellow opponent, which is described in detail in Würtz and Lourens [44,45,22].

The amplitude for each of the red-green, blue-yellow, and grey value channels yields the final edge operator

$$\mathcal{C}^{\text{all}}_\sigma = \sqrt{\mathcal{C}^2_\sigma + {\mathcal{C}^{(r,g)}_\sigma}^2 + {\mathcal{C}^{(b,y)}_\sigma}^2} \quad (10)$$

at a single scale. Since $\mathcal{C}^{(e,i)}_\sigma = \mathcal{C}^{(i,e)}_\sigma$ one channel for every opponent pair is sufficient. Similarly the final corner operator at a single scale is as follows:

$$\mathcal{E}^{\text{all}}_\sigma = \sqrt{\mathcal{E}^2_\sigma + {\mathcal{E}^{(r,g)}_\sigma}^2 + {\mathcal{E}^{(b,y)}_\sigma}^2} \ . \quad (11)$$

4.5 Combining Multiple Scales

Sharp corners are characterized by strong responses over a wide frequency range. If only high frequency cells respond, the feature is likely to be noise or texture rather than a corner. We found that averaging the responses over a range of frequencies yields a much more robust corner detection [44,45]:

$$\mathcal{E}^{\text{all}}_{\text{avg}}(x,y) = \frac{1}{S}\sum_{j=0}^{S-1} \mathcal{E}^{\text{all}}_{\sigma_j}(x,y) \ , \quad (12)$$

where S denotes the number of scales.

Also the complex responses are combined by averaging over the $\lceil S/2 \rceil$ smallest scales:

$$\mathcal{C}^{\text{all}}_{\text{avg}}(x,y) = \left\lceil \frac{2}{S} \right\rceil \sum_{j=0}^{\frac{S}{2}-1} \mathcal{C}^{\text{all}}_{\sigma_j}(x,y) \quad . \tag{13}$$

Combining the smallest scales only, yields better results for contour extraction. Extraction of contours (or corners) is difficult when a receptive field influences two or more contours (or corners) at the same time. This is more likely to happen for contours than corners because they are one-dimensional features, while the latter are zero-dimensional. Contour extraction will be described in Sect. 5.

5 Graph Extraction

Representing extracted edges and corners as attributed graphs is recognized to be useful for object recognition by graph matching [3,6,8,23,24,25]. Before object recognition can take place corners and contours (edges) need to be extracted.

For edge a contour following algorithm will be proposed that is based on similar principles as the border-tracking algorithm [10], and following as graph searching [1]. The main difference of the algorithm that will be proposed is the data (enhanced contours, detected corners, and local contour (edge) maxima) that will be used for extraction of contours.

Many approaches of edge detection are based on strong mathematical formalisms (e.g., Lindeberg [21] uses n-th order partial derivatives of a Gaussian kernel combined with scale space; Canny [4] tries to maximize the gradient) rather than on human perception of edges. In most methodologies several parameters, like threshold and length of contour need to be set. Also contours are not guaranteed to be without gaps. The latter is essential for object recognition by graph matching.

Automating the thresholding of edges is done by Rosin [35,43] based on the central limit theorem. Although results are promising, there is no evidence that contours are without gaps, neither that detected edges are also perceived by humans as edges. Approaches for edge detection based on maximum phase congruency [27] avoid the issue of thresholding, later Venkatesh and Owens [42] used the local energy function to calculate phase more conveniently. Recently Kovesi [20] improved this method by using wavelets. Although the method shows very good results and avoids thresholding, it is ill defined at low contrast and contours often contain small gaps.

Difficult settings like length and threshold can be avoided easily if segments are extracted by contour following, instead of performing the operations on the image. Another advantage of contour following is that segments are without gaps, otherwise the contour is not extracted. The idea of contour following is based on walking from one mountain top to another. During the walk to the other top one should try to stay has high as possible. This means that we select corners and local maxima as starting points and then to follow a contour to another corner or local maximum by selecting the strongest edge responses.

Globally the algorithm contains 5 stages:

1. detecting corners and local edge maxima by thresholding
2. extracting contours starting at local maxima
3. extracting contours starting at corners
4. connecting corners to the extracted contours
5. constructing a graph with additional arguments from extracted corners and contours

The choice for a threshold in most algorithms strongly depends on the input image. We created a robust corner operator by averaging over multiple scales, hence setting one threshold at a reasonable value yields good results in almost all images. Selection of local edge maxima is a less critical process than marking corners, therefore a constant threshold at a single scale that is lower than that for marking corners gives results that are satisfactory. Marking corners and local edge maxima will be described in more detail in Sect. 5.1.

Contour extraction is done in the second and third step of the algorithm. The steps are done separately because local edge maxima and corners are different features and consequently will be treated differently. At a corner several, but at most eight, contours can start (or end), and in order not to miss any, a contour will be followed in every direction. At a local maximum, contours will be followed in two *opposing* directions. The directions should be opposing otherwise a corner would have been detected at that position.

Contours do not necessarily have to pass through a corner, this is the case for, e.g., a rounded corner. To get a fully symbolic representation, which is needed for symbolic reasoning, an additional step is taken that connects

```
 1 Function ExtractGraph (E, C, σ)
 2     C := Set of corners obtained from E
 3     M := Set of local edge maxima obtained from C
 4     forall m ∈ M
 5         forall n ∈ BestNeighborsSelectedByResponseAndOpposite (C, m)
 6             l := ExtractContour (C, C, M, m, n)
 7             Add (L, l)
 8     forall c ∈ C
 9         forall n ∈ EightNeighbors (c)
10             l := ExtractContour (C, C, M, c, n)
11             Add (L, l)
12         RemoveDoubleDetectedContours (L)
13     forall l ∈ L
14         forall c ∈ C
15             l_c := ConnectCornerToContour (l, c, dσ)
16             Add (L_c, l_c)
17     G := CreateGraph (L ∪ L_c)
18     return G
```

Fig. 4. Algorithm for graph extraction.

corners to contours if the closest distance between corner and contour is less than $d\sigma$. The value $d\sigma$, is the distance where the complex cell operator influences the end-stopped operator strongly by definition of the single end-stopped operator (3). Details about graph extraction will be elaborated on in Sect. 5.2.

The fifth step in the algorithm is the construction of a graph $G = (V, E)$, where V is a set of vertices representing corners and E is a set of edges. In normal graphs an edge is represented as a pair of vertices. In our case also the contour itself (a chain of pixels encoded by freeman chain indices), the length of the contour (in pixels), and the average $\mathcal{C}^{\text{all}}_{\text{avg}}$ response of the contour are included.

5.1 Marking Corners and Local Edge Maxima

A position (x, y) is marked as a *corner* if the $\mathcal{E}^{\text{all}}_{\text{avg}}(x, y)$ response is larger than its neighbors and above $T_{\mathcal{E}}$. Similarly (x, y) is a *local maximum* if $\mathcal{C}^{\text{all}}_{\text{avg}}(x, y)$ is larger than its neighbors and above $T_{\mathcal{C}}$.

For detecting corners and local edge maxima thresholds need to be set. We choose these thresholds such that most noise and small artifacts are avoided. For corner detection the threshold is set to $T_{\mathcal{E}} = 20$ this will average out all texture and noise responses, although corners with contrast smaller than 30 pixels will not be detected.

For selecting local maxima we choose $T_{\mathcal{C}} = 10$ which is about a factor two higher than what human beings can discriminate.

Note that thresholds $T_{\mathcal{E}}$ and $T_{\mathcal{C}}$ are found empirically, but the same for all used input images!

5.2 Algorithm for Graph Extraction

This section describes the algorithm of symbolic graph extraction from enhanced edges (13) and corners (12). In lines two and three of the graph extraction algorithm (Fig. 4) a set of detected corners and local edge maxima are created as described Sect. 5.1. From a local maximum we start following the contour in two opposite directions. The directions are selected by taking the coordinate from the eight (connected) neighbors that has the highest $\mathcal{C}^{\text{all}}_{\text{avg}}$ response while the other coordinate is opposite to this selected coordinate (line 5). The algorithm for following, including stop criteria is given in Fig. 5, will be described below.

At a corner two or more contours end, hence contour following is done in all possible (eight) directions (line 9). Since it is not clear how many contours start from the corner an additional step needs to be taken to avoid doubly detected contours (line 12).

Contours do not always pass through a corner exactly. An additional step is needed to connect corners to extracted contours that lie within distance $d\sigma_{\text{avg}}$ ($\sigma_{\text{avg}} = \sum_{i=0}^{S/2-1} \sigma_i$ is the average scale) from each other. In case a corner is within this distance the shortest path from corner to contour will form a new segment (line 15).

Finally, all segments are then used to construct a graph (line 17).

The idea for contour following is based on walking from one top, local edge maximum, to another and trying to keep as high as possible. The latter is done by selecting always that coordinate (within certain constraints) with

```
1  Function ExtractContour (C, C, M, c n)
2      p := n; segment := p; pp := c
3      repeat
4          if (NotAtBorderImage(p) and NoMember(segment, p) and C(p) ≥ T and p ∉ C ∪ M)
5              p1 := neighbor of p to pp at 180°
6              p2 := neighbor of p to pp at 135°
7              p3 := neighbor of p to pp at 225°
8              pp := p
9              p := pj, where C(pj) = max(C(pi)) ∀i; i, j ∈ {1, 2, 3}
10             Add (segment, p)
11         else stop
12     until stop
13     if (p ∈ C ∪ M or Length (segment) ≥ L)
14         return segment
```

Fig. 5. Algorithm for extraction of a single contour from local edge maximum or corner to another local edge maximum or corner.

the highest $\mathcal{C}_{\mathrm{avg}}^{\mathrm{all}}$-response. The algorithm is given in Fig. 5. In line 2 the initialization of the algorithm is done; p denotes the current point or coordinate and pp previous point which is initially a corner or local edge maximum. Parameter *segment* is a chain of coordinates that describe the extracted contour. If all criteria in line 4 are satisfied, a new coordinate will be selected. At every step there are three possible neighbors of p that can be selected. These neighbors form a 135, 180, and 225 degree angle with p and pp. From these three neighbors, that one is selected, that has the highest $\mathcal{C}_{\mathrm{avg}}^{\mathrm{all}}$-response (line 9). After selection previous point pp is set to p, while current point p is set to the neighbor with the highest response. The algorithm stops if p is at the border of the image, p is visited before in the current segment, the response is below a threshold T which is chosen to be extremely low (usually 1), or when another local maximum or corner is found. Note that in case p is nor a corner neither a local edge maximum the contour will be kept if it has sufficient length ($L \geq 15$).

5.3 Results of the Algorithm

The results of the algorithm are illustrated in Fig. 6. In total seven corners are detected (white squares with black borders), here one corner, at the bottom of the stalk of the P, is not immediately perceived as corner. When the image is viewed from a longer distance three corners are perceived only, among them is this corner (line end). The P-image illustrates that corners only are not sufficient to extract all edges. Local edge maxima are needed in case of circular contours.

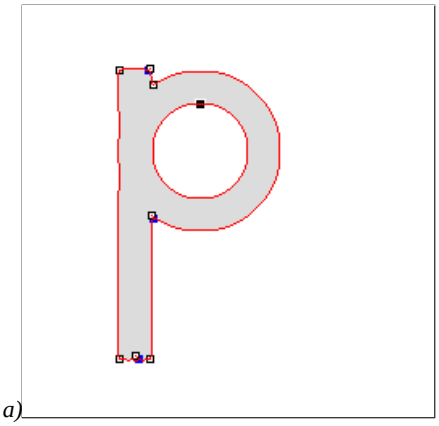
In an additional step the corners are connected by the shortest distance to the curve that are on a distance smaller than $d\sigma_{\mathrm{avg}}$. Connection points between corner and contour are given by dark squares. The extracted graph (Fig. 6b) contains 11 vertices and 11 contours. The vertices include 7 corners (white squares with black borders) and 3 points (dark squares all partially occluded by a corner) to connect the corners to the contours and one local edge maximum (black square with white border) to mark the starting and ending point in a closed contour.

6 Object Recognition by Graph Matching

For finding subgraph isomorphisms between model (object) and image graph we have modified the algorithm by Ullman [41] based on tree search with backtracking. To cut down evaluation expenses attributes (Sect. 6.1) are assigned to vertices and edges. To cut down evaluation expenses in graph matching often only the best matching copy is searched. This is not acceptable here, because the same object may appear several times in the image graph. Consequently, we require to find *all* copies of a model graph G_m in the given image graph G.

In this section we will introduce different attributes that make graphs invariant under translation, rotation, and scale. A cost function and a-priori known similarity thresholds (Sect. 6.3) are used to make object recognition flexible and fast, although this problem is known as being NP-hard. In Sect. 6.4 we will present and explain the graph matching algorithm.

For the reader interested in related work, technical details, time complexity and performance of object recognition by inexact graph matching for real world images we refer to Lourens and Würtz [23,24].



a)

b)

V_i	V_j	(x_i, y_i)	(x_j, y_j)	len	score
1	2	(60, 40)	(78, 40)	17	173.39
3	6	(81, 49)	(81, 132)	197	180.60
2	**3**	(78, 40)	(81, 49)	8	152.38
4	*4*	(110, 61)	(110, 61)	167	179.53
8	**1**	(60, 219)	(60, 40)	179	187.12
10	9	(79, 219)	(72, 219)	6	164.39
6	**10**	(81, 132)	(79, 219)	86	186.15
9	**8**	(72, 219)	(60, 219)	11	174.13
5	6	(80, 130)	(81, 132)	1	136.24
7	9	(70, 217)	(72, 219)	1	145.05
2	**0**	(78, 40)	(79, 39)	0	***

Fig. 6. Extracted contours of P input image (Fig. 3a). *a)* Graphical representation. Marked corners (white squares with a black border), local edge maximum (black square with a white border), connecting points (dark squares all partially occluded by corners), and extracted contours (dark grey). *b)* Symbolic representation of the extracted graph. Corners are represented in boldface and the local maxima are emphasized. Note that the extracted chain is omitted. The used parameters are $T_{\mathcal{E}} = 20$, $T_{\mathcal{C}} = 10$, $T = 1$, and $\sigma_{\text{avg}} = 3.24$.

6.1 Graph Attributes

After contour following, we end up with a graph that has corners (local edge maxima and connecting points) as vertices and contours described by a series of points as its edges. Once stored models and the image to be analyzed are represented in this way, model matching can be done by finding a copy of the model graph in the image graph.

Each corner is labeled with the angles between all pairs of adjacent line segments starting from it. The edges are labeled with the *relative length* of the line segments, i.e., the ratio of the length to the length of the longest line segment in the whole graph.

The extracted corners are known by their (x, y)-coordinates. To make a graph *translation invariant* these are not directly suitable for attributes, but only the *relative* positions may be used. This is obtained by taking the *lengths of the contours* and the starting angles between contours at corners. This makes the representation *rotation invariant* as well. Finally, to make the graph scale independent we choose the lengths to be relative to each other.

6.2 Cost Function

The standard representation of a graph is the Boolean adjacency matrix B. Its elements $B(i, j)$, $i < j$ and $i, j \in V$, of this matrix are chosen to be true if and only if $(i, j) \in E$, and false otherwise. We generalize it by replacing the Boolean with a real-valued *cost function* $EC(v_1, v_2)$, $v_1, v_2 \in V$, which defines the cost of an edge missing in the image graph. During the process of matching we sum the costs for these tolerated missing edges. This accommodates cases where missing edges in certain places are considered less severe than in others. Since we use undirected graphs we only need the lower- or upper-diagonal of the adjacency matrix.

To cut down evaluation costs during matching we try to evaluate as little as possible, while allowing inexact matches and without losing any solutions. During the matching process we tolerate one or more missing edges in the image graph. Only inexact matches with a total cost less than an a-priori known cost are considered.

The flexibility of the edge matching reflects the philosophy that contours are often weak in real images, so contour following may fail in the image. It is, however, assumed that the model is known well enough, i.e., images of sufficient quality are available, for all relevant contours to be present in the image graph. Beside the added robustness this flexibility yields a method of processing unconnected model graphs by adding extra edges.

6.3 Attribute Similarity Thresholds

Vertices are augmented with a set of angle and ratio attributes. In the matching process these attributes are used to reduce the search space. When a vertex is matched we tolerate an angle and a ratio at the vertex in the image graph to differ by at most a known constant angle and a known constant ratio from the corresponding angle and ratio in the model graph. Additionally we allow the average angle difference of all angles to differ at most a known constant angle. Similarly the average ratio difference is also bounded by a constant. We introduce four bounding constants:

```
 1 Procedure MatchGraph (G, MG)
 2     stack := ∅
 3     forall v ∈ V(G)
 4         L_0    := v                    /* list of parsed vertices of image graph */
 5         ML_0 := first model vertex     /* list of parsed vertices of model graph */
 6         mv    := 1                     /* number of matched vertices             */
 7         cv    := first model vertex    /* vertex being evaluated                 */
 8         cva   := first angle of cv     /* angle of cv to be evaluated            */
 9         ED    := 0.0                   /* accumulated edge difference            */
10         AD    := 0.0                   /* accumulated angle difference           */
11         Push (mv, cv, cva, L, ML, ED, AD)
12     while stack ≠ ∅
13         Pop (mv, cv, cva, L, ML, ED, AD)
14         if mv = cv = #V(MG)            /* Match found                            */
15             Evaluate maximum and average relative length differences
16         else
17             if Angle cva of vertex cv can be evaluated
18                 if Evaluation accepts angle and ratio
19                     if cva = last angle of cv
20                         cv := next (cv)
21                         cva := first angle of cv
22                     else
23                         cva := next (cva)
24                     Push (mv, cv, cva, L, ML, ED, AD)
25             else                       /* Add a missing vertex                   */
26                 ML_mv := missing-vertex
27                 forall v ∈ V(G) − L    /* Unused vertices only                   */
28                     L_mv := v
29                     if ProperVertex (EC, max ED − ED)
30                         Push (mv + 1, cv, cva, L, ML, ED + EC, AD)
```

Fig. 7. Algorithm for graph matching. See Sect. 6.4 for explanation.

1. the angle tolerance,
2. the average angle tolerance,
3. the ratio tolerance, and
4. the average ratio tolerance.

We assume that the model graphs are either constructed by hand or extracted from "clean" images. Thus, they are supposed to contain *all* edges and corners.

6.4 Graph Matching Algorithm

The algorithm for graph matching is illustrated in Fig. 7. It can find all copies of a model graph G_m in an image graph G. Lines 2–11 are the initial stage of the algorithm, we start with an empty stack and push all vertices of the image graph onto the stack, one after another, since each of them can, in principle, be matched with the first vertex of the model graph. Lines 12–30 constitute the matching proper. In line 13 we take a possible partial solution from the top of the stack and check if the match is already complete (line 14). If so, the maximal and average relative length differences to the model are calculated; if both are within the bounding constants the match is accepted and displayed. If the match is incomplete the process continues at line 17. Here we check if the current angle and ratio can already be evaluated. If we can not evaluate because one or both vertices to form the angle are still missing, then the missing model vertex is parsed by adding it to the list (line 26) and all possible vertices in the image graph (lines 27–30) are searched. A vertex v is added if it is not yet matched and if the cost EC of adding edge (L_{cv}, v) is smaller than the allowed cost.

The speed of the algorithm mainly depends on the condition in line 18. If angle and ratio differences are chosen properly most of the partial matches will be rejected here, and the path is rejected by not pushing it back on top of the stack. During matching, the difference in ratio $\delta r = \max\left(\frac{r_m}{r}, \frac{r}{r_m}\right)$ (where r_m denotes the ratio of an edge pair in the model graph and r denotes the corresponding matched ratio in the image graph) between two pairs of

edges is obtained by scaling the edge pairs in such a way that the first edge of both pairs is one, then the ratio is the size of the rescaled second edge of the first pair: the size of the rescaled second edge of the second pair. The average length difference is evaluated by using the relative lengths of both model and found match in the image graph. The absolute difference of the model edge with its corresponding image edge is taken, when there was no edge between v_a and v_b in the image graph we used the relative length of dist(v_a, v_b). The average of all edges is taken to represent the average relative length difference.

6.5 Stereo Matching

Stereo matching by using graph matching consists of two stages. Firstly, we apply the graph matching algorithm twice, once to the extracted graph from the left camera image and once for the graph from the right camera image. We assume all known (modeled) objects in both images are recognized.

Secondly, we apply the algorithm once more to recognize the corresponding object pairs. We use the recognized objects (of the first stage) in the left image as a set of model graphs and the recognized objects in the right image as image graph. From the resulting pairs we take the camera (x, y) coordinates to extract depth.

The current stereo implementation assumes that

1. all objects are recognized at both stages
2. the same object appears only once in the scene

In constructing a scenario we consider the above two limitations.

7 Sound

In the system, sound signals are sampled at 48 kHz using two pairs of binaural microphones. After sampling, the input sounds are transformed into spectrograms by a fixed incremental time step (7.5 ms) using FFT for 1,024 points. The frequency analyses of the four channels are processed synchronously to keep information between channels such as *IPD (interaural phase difference)* and *IID (interaural intensity difference)*.

After noises are canceled on created sound spectrograms, the system currently extracts two features: pitch and direction information.

7.1 Noise Cancellation

The humanoid robot which combines audition with motion, i.e, *active audition*, causes noises because it is inevitable that motors make noises while in motion [29]. In addition, such noises are captured relatively loud because the microphones are located relatively near the motors of the robot. Therefore, noise cancellation is indispensable for active audition. To cancel motor noises, a cover is used to separate the inner from the outer world. Hence one pair of microphones is installed at the inside of the cover at the ear position of the robot, while the other pair of microphones is installed very close to the former ones on the outside of the cover. This enables effective noise cancellation by using an adaptive filter by comparing internal and external sounds.

Our adaptive filter uses *heuristics with internal microphones*, which specifies the condition to cut off burst noise mainly caused by motors: for example, when the body hits some stoppers to ensure safe movement. The heuristics orders that localization by sound or direction-pass filter ignore a subband if all of the following conditions hold:

1. The power of internal sounds is much stronger than that of external sounds,
2. Twenty adjacent subbands have strong power (30 dB), and
3. The robot is moving, i.e. one or more motors are active.

7.2 Pitch Extraction

In the system, a pitch is represented as a set of *frequency components*, which have harmonic relationships with one another. A frequency component is a series of peaks in the time direction in a spectrogram. A *peak* is defined as a local maximum in a spectrum. A frequency of pitch is called a *fundamental frequency* when it corresponds to the lowest frequency of components in a pitch.[1] Two important components of pitch are *onset* and *offset*. They indicate the earliest start time and the latest end time of all frequency components in a pitch, respectively.

In pitch extraction, accurate peak extraction is one of the most important problems because each peak extracted from left and right channels is used to determine a reference pair. These pairs are used to calculate the

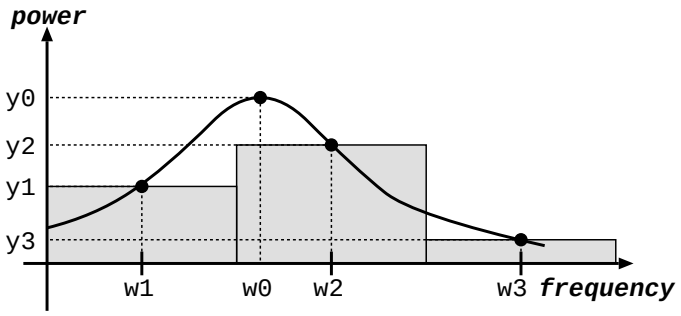


Fig. 8. Peak extraction.

direction of the sound sources. Since direction can be used for tracking a target in both audition and vision, finding corresponding peak pairs is an important problem.

Our peak extraction method is based on spectral subtraction [2]. It uses peak approximation method based on characteristics of FFT and window function. Consider that a spectrum contains the following $[\omega_1, y_1]$, $[\omega_2, y_2]$, and $[\omega_3, y_3]$ tuples, generated by FFT as shown in Fig. 8. Where the first parameter ω_i denotes the frequency and the second one y_i represented by a complex number is the power. Primarily ω_2 is detected as local maximum, however due to discretization this may not be the true peak. By interpolation peak $[\omega_0, y_0]$ is estimated as follows:

$$\omega_0 = \begin{cases} \omega_2 - \frac{2\pi(-|y_2|+2|y_3|)}{T(|y_2|+|y_3|)} & \text{if } \omega_2 < \omega_0 < \omega_3 \\ \omega_2 + \frac{2\pi(2|y_1|-|y_2|)}{T(|y_1|+|y_2|)} & \text{if } \omega_1 < \omega_0 \leq \omega_2 \end{cases} \tag{14}$$

$$\mathrm{Arg}(y_0) = \mathrm{Arg}(y_2) + \frac{T}{2}(\omega_2 - \omega_0) \tag{15}$$

$$|y_0| = \frac{T^2\delta_\omega\left(-\delta_\omega^2 + 4\pi^2/T^2\right)|y_2|}{2\pi^2 \sin\frac{T}{2}\delta_\omega} \ , \tag{16}$$

where $\delta_\omega = \omega_2 - \omega_0$ and T is the window length of the FFT. Frequency ω_0 is estimated by (14), while phase and amplitude of y_0 are estimated by (15) and (16), respectively.

The above proposed method is more accurate and 200 times faster than Bi-HBSS, which uses a similar peak extraction method.

7.3 Sound Source Localization

In general, the head related transfer function (HRTF) can be used to obtain the direction from IID and/or IPD. However, HRTF is sensitive to changes of the environment. Whenever the environment changes even a little, re-measurement of HRTF is required. Therefore it is difficult to apply a localization method using HRTFs to mobile objects such as robots.

We propose a sound source localization method based on *auditory epipolar geometry* instead of HRTF [29]. We obtained the idea of epipolar geometry from stereo vision, where it is a general method, and transfered it to the auditory field.

Let $y_0^{(r)}$ and $y_0^{(l)}$ be corresponding peaks from right and left channel, respectively. Then the IPD $\Delta\varphi$ is calculated as follows:

$$\Delta\varphi = Arg\left(y_0^{(r)}\right) - Arg\left(y_0^{(l)}\right) \ . \tag{17}$$

Angle θ denotes the horizontal angle, which rotates counter clock wise around the z-axis and where 3 o'clock corresponds to 0 degrees. It is calculated as follows:

$$\cos\theta = \frac{v}{\omega_0 b}\Delta\varphi \ , \tag{18}$$

where v is the velocity of sound and b is the distance between left and right microphones. For the moment, the velocity of sound is fixed to 340 m/sec and remains the same even if the temperature changes.

[1] Pitches without fundamental frequency exist. We call these exceptional pitches as *missing fundamentals*.

8 Experimental Setup

For the interaction between audio and visual information, a humanoid robot is used, designed by Kitano et al. [18]. The humanoid torso has four degrees of freedom (DOF) for the body which is driven by four DC motors each controlled by a potentio meter. Two Sony EVI-G20 CCD cameras are used each with three DOF for pan, tilt, and zoom to form the visual input. Two pairs of nondirectional Sony ECM-77S microphones are used, to supply the model with audio input.

A relatively simple scenario in a real world environment is set up for the robot, to illustrate that audition and vision in certain cases benefit from each other. Two targets are used:

- a rectangular two-dimensional object attached to a sound source with a low pitch and
- a triangular two-dimensional object attached to a sound source with a high pitch.

The visual objects are chosen to be two-dimensional and with clear corners and edges in the scene, this will ensure proper form extraction, recognition, and stereo matching. Similarly we selected two clearly distinguishable sound sources also to ensure proper localization and recognition of these sources.

The aim of this experiment is not to show that we can handle complex sceneries in audition and vision, but that these cues can benefit from each other.

8.1 Object Recognition

A knowledge database $T = S \cup O$ containing $S = \{s_i\}, i = 1 \ldots N_s$ sound particles and $O = \{o_j\}, j = 1 \ldots N_o$ visible objects is initially available.

In the experiment a database contains a description of a triangle (o_1) and a rectangle (o_2). All visible objects are described by form features only, i.e. corners and the connectedness of the corners by edges. Further two sounds are in the database s_1 and s_2 for a high (525 Hz) and low (262 Hz) pitch, both with harmonics structure. The sounds are described by pitch and direction.

In our scenario no new objects will be introduced, hence modules *Ask teacher*, *Update knowledge*, and *Add new knowledge* in Fig. 2 are not actively used.

For visual object recognition, we used the graph matching algorithm described in Sect. 6.4. The algorithm uses two input graphs: $G = (V, E)$ and $G_{o_j} = (V_{o_j}, E_{o_j})$ to represent the current scene and all objects o_j. The algorithm finds all inexact matches of objects o_j in the visual scene represented by G.

8.2 Active Vision and Audition

The input data of the humanoid is huge. Images are captured at a frame rate of 30 Hz in NTSC format, while sound is captured at 48 kHz.

We are currently not able to process the proposed algorithms, mentioned in previous sections, in real time. Hence we create simple preprocessing mechanisms that generate events. Only when an event occurs the data is processed.

In vision an event occurs

- at the first frame
- after the robot stops moving
- when the robot is not moving and a frame differs considerably from the frame of the previous event

In audition an event occurs

- when sounds are clearly above the noise level by using increasing power and attack time to get a so-called *onset* event
- when the power decreases to the noise level an *offset* event is generated, where an offset event always occurs after an onset event.

We realize that events generated by both cues have a lot of shortcomes. For example the threshold value used for the summed difference between the pixel values of two frames is critical. At a low threshold value, noise can generate an event. At a higher threshold value small moving objects might be missed. For audition, event generation is not very useful if the volume is all the time above the noise level, after an onset event is generated. Currently, at a vision event we do not localize the position of the differences to anticipate where a moving target might be, and subsequently only process the data locally.

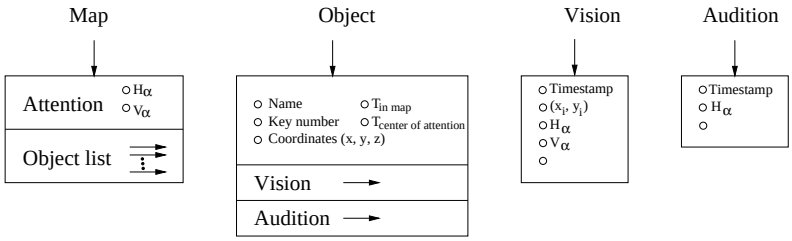


Fig. 9. The data structure of our map contains global attention coordinates (H_α, V_α) and a set of objects. A single object contains a name, key-number to let every object in the scene be unique, global 3D coordinates for the center of the object, and parameters used for attention. The object also contains vision and audition information. Both vision and audition contain a time stamp, local attention coordinates, and in case of vision also a set of 2D corner coordinates for left and right image to describe the object compactly.

8.3 Map

The map, that can be interpreted as "memory" of the humanoid, contains information about all recognized objects over time. The map is a dynamic data structure that is contains two parameters H_α and V_α for the focus of attention, and a set of objects, which is initially empty. During a simulation different recognized objects will be stored in this set.

A single object contains information about vision and audition, as well as several parameters. The object is localized by 3D coordinates and identified by name and key-number (Fig. 9). For the focus of attention we used two parameters: the time that the object was recognized and when it was for the last time the center of attention.

The vision parameters are available only when the object can be seen. When visible, current time, localization, and all local corners extracted from left and right camera images are stored as 2-D coordinates. For audition, only current time and localization parameters are used. In our experiment we assume that all objects are known, hence these vision and audition parameters are currently sufficient.

8.4 Selective Attention

The output of the object recognition module (see Fig. 2) gives all possible objects RO in the database that changed or are new compared to the previous map. These recognized objects form the input for the selective attention model; see algorithm in Fig. 10. In case all objects are known in the map, attention might be updated only.

Initially at $t = 0$ the map is empty, i.e. the humanoid has no information about the scene. In that case the attention is set in the center part of the visual scene with low attention value to indicate weak attention (line 3 of Fig. 10). Lines 4-8 update or add information to the map. For every object an initial value for attention is given. In Line 9 the attention is updated. The final focus of attention is determined by the highest attention value. For this value an evaluation function is used that includes the time the target has been in the map, and how long the target was center of attention. Lines 10 and 11 extract and return the features from the map, respectively.

8.5 Scenario

In the scene a flute and a violin, represented by triangle o_1 and sound s_1, and rectangle o_2 and sound s_2, respectively, are used.

The following scenario is played: At time stamp $t = 0$, the violin is visible in the right part of the scene; the flute is initially invisible. The object is recognized and the robot is instructed to focuss on the target by moving to the right and bending the head a little forward. At the time the robot stops moving the violin starts playing and events occur for both audition and vision (time stamp $t = 1$). When the violin stops playing we move it to the left. The image data which is produced constantly by the frame grabber differs significantly and a new event occurs. The object is recognized and the robot shifts its attention to the left by rotating the body. Since the object is moving constantly also events occur frequently; we marked them by intermediate time stamps 1.1, 1.2, ..., 1.n. When the violin stops moving the last event of the moving sequence occurs, which we marked $t = 2$. Then the flute starts playing, which is invisible, an audition event occurs ($t_a = 3$). After processing the data and extracting the direction, the robot moves to the left. After the robot stops a new event occurs for vision since the current image differs from the previous time stamp. After visual recognition of the flute the position can be determined more

```
1  Function SelectiveAttention (t, RO)
2      if t = 0
3          Initialize (Map = ∅, Attention = [center; weak])
4      forall RO_i
5          if InMap (Map, RO_i)
6              UpdateMap (Map, RO_i, Attention=[Coord(RO_i); medium])
7          else
8              AddToMap (Map, RO_i, Attention=[Coord(RO_i); strong])
9      UpdateAttention (Map, Attention, RO)
10     F = ExtractFeatures (Map)
11     return F
```

Fig. 10. Algorithm for selective attention. Input parameters: t is timestamp and RO is the set of recognized objects in current scene that differed from the results in the previous map at the previous time stamp.

accurately and the robot focuss its attention to the center of the flute by moving again, after motion a new vision event occurs ($t_v = 3$). The flute stops playing and the violin starts moving again (to the left). When the violing is sufficiently visible a new event occurs ($t = 4$). After processing the robot updates its attention ($t = 5$). Finaly we play the flute again and move attention back to the flute ($t = 6$).

From the moment ($t = 4$) that both objects are in the scene and attention might change if one or more of the following actions occur:

- hearing a sound,
- detecting a moving target, or
- by attention to a new object in the scene.

By playing the above scenario we show that audition is needed, since in the beginning the flute is invisible and attention is attracted when the flute starts playing. Only two objects are in the scenario, that indicates that once they are recognized, stored in the map, both are visible, and do not move, object recognition is not needed anymore, which speeds up the model considerably.

9 Results

In Fig. 11a and c, 11b and d, and 11c and f, an example of input data, an intermediate level of feature processing for vision and audition, and fully processed features are given, respectively. The latter from the input for the attention model illustrated in Fig. 2.

Figure 12 illustrates the left images of interesting events used in the scenario. At $t = 0$ initialization takes place, the z-axis is determined by the initial focus of attention. The global coordinate system is fixed, where (x, y, z) represent horizontal vertical, and depth axis. Initially the violin is recognized directly. At $t = 1$ the attention is at the violin because it is recognized and playing. At $t = 2$ the violin moves 50 centimeters to the left, attention is still at the violin. At $t = 3$ the flute starts playing and is purely recognized by the sound because it is outside the visual field (see Fig. 12, $t = 2$). At $t = 4$ the violin again moves another 40 centimeters to the left and at $t = 5$ attention has changed to the violin because it moved. Finally the flute starts playing again and attention is changed to the flute again.

Table 1 gives the results of the map represented by a whole row (attention included) at different timestamps. Attention is given by H_α and V_α which denote the rotation in degrees about vertical (elevation) and horizontal (azimuth) axis, respectively. The angle of the depth is omitted since the stereo setup is in cannonical form. It can be easily derived from Table 1 which object is focussed at, since their (x, y) coordinates are around the origin.

In the same table, the violin and flute are represented by four and three corners, respectively. They are represented by local coordinates. Although these coordinates can be easily converted to the global (world) coordinate system by using the attention parameters.

Differences in calculated and measured coordinates are due to resolution, possible drifts of a recognized corner by one or two pixels, or by inaccuracy of the attention due to small errors in the motor control. For instance if a recognized corner shifts one pixel in horizontal direction on the CCD, it implies a shift in depth of 8 centimeters when the object is at about two meters distance. Depth can be obtained only if an object is less than six meters away from the camera otherwise there is no shift between left and right CCD image.

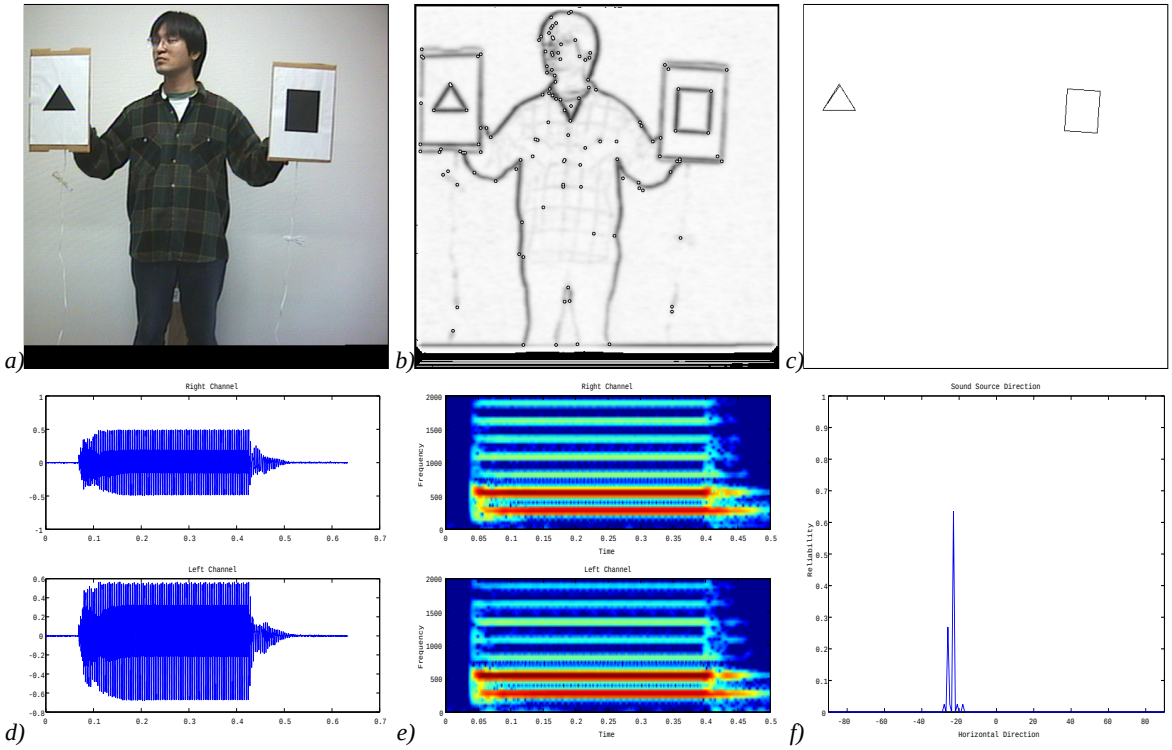


Fig. 11. An example of feature extraction and recognition. *a)* Visual input image. *b)* Form features: enhanced edges and recognized corners. *c)* Recognized objects: black triangle and square. *d)* Audio input data. *e)* Intermediate results of sound processing given by a spectrogram. *f)* Recognized horizontal direction of the sound source.

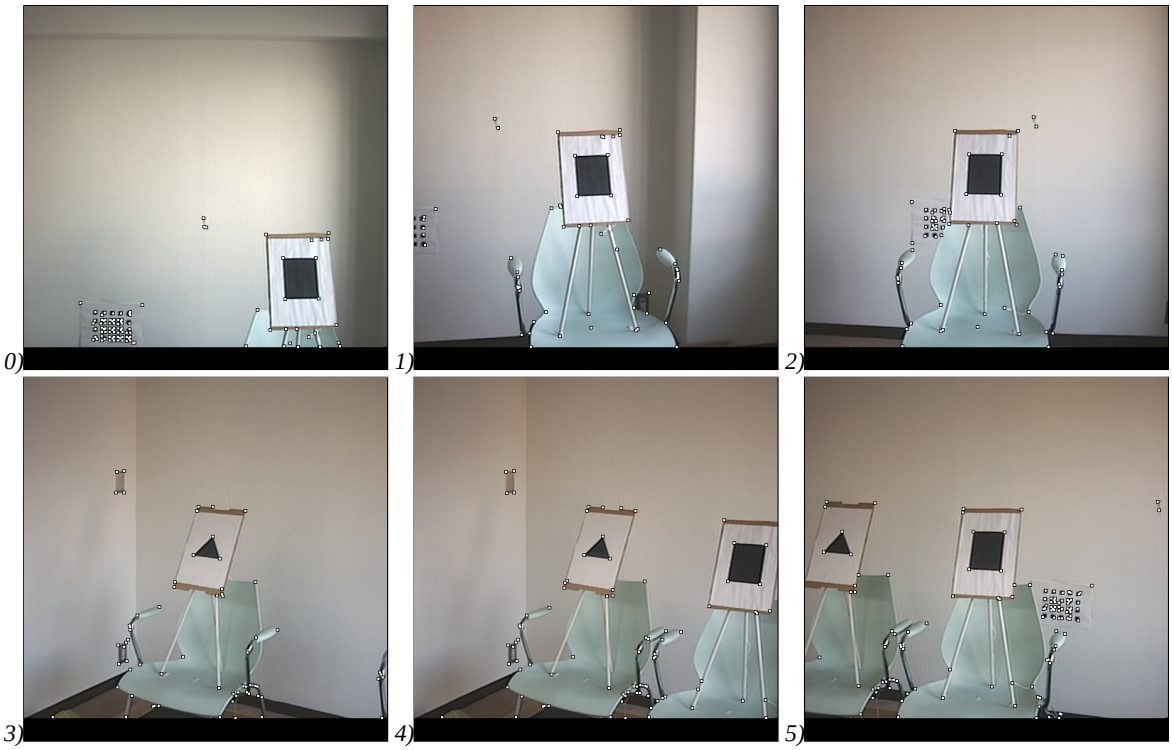


Fig. 12. Illustration of sceneries of interesting events of the played scenario, images from the left camera at different timestamps are given. In all images recognized corners are marked as white squares with black edges and put on top of the original images. Note that $t = 6$ is not illustrated because it is similar to $t = 4$. The attention is at the center of every image. Its anglular coordinates in the map are given in Table 1 for the different timestamps.

Table 1. Results of the selective attention model. At every timestep the most relevant elements of the map, given by one horizontal row (except the last column), are updated. Local attention from vision and audition are given by H_α and V_α, which denote the rotation in degrees about vertical and horizontal axis, respectively. Differences between audition and vision are due to accuracy of audition. Which indicates similarities between human audition who have an accuracy in horizontal direction of $\pm 8^\circ$ [5]. In the leftmost column, different timestamps are given. The results of the violin (rectangle represented by four corners) are presented in the first two rows, and (when visible) the results of the flute (triangle represented by three corners) are presented in the last two rows. In the map the centers of the recognized objects are given which are obtained from the local coordinates. For comparision the measured data is illustrated in the rightmost column.

		local 3D coordinates		sound	global 3D coordinates in cm	
t	object	corner coordinates in cm	attention	direction	object center	measured
0	violin	(22, -28, 185) (35, -29, 191)	H_α= 0		(24, -37, 178)	(27, -38, 183)
		(23, -41, 185) (36, -43, 191)	V_α = 0			
1	violin	(-11, 7, 179) (0, 7, 185)	H_α= 8.8	H_α= 12.8	(-24, -37, 177)	(-23, -38, 183)
		(-11, -7, 191) (1, -6, 179)	V_α =-11.1			
2	violin	(-10, 7, 179) (1, 8, 185)	H_α= -5.9		(-24, -37, 177)	(-23, -38, 183)
		(-11, -6, 185) (1, -6, 179)	V_α =-11.1			
3	violin				(-24, -37, 177)	(-23, -38, 183)
	flute	(-1, 4, 205)	H_α=-31.9	H_α=-35.3	(-115, -45, 169)	(-123, -39, 183)
		(-9, -3, 213) (1, -4, 205)	V_α =-11.4			
4	violin	(45, 2, 191) (56, 1, 191)			(-62, -45, 170)	(-63, -38, 183)
		(43, -12, 191) (54, -13, 191)				
	flute	(-1, 4, 213)	H_α=-31.9	H_α=-35.3	(-115, -45, 170)	(-123, -39, 183)
		(-8, -3, 205) (1, -4, 205)	V_α =-11.4			
5	violin	(-9, 6, 191) (3, 6, 191)	H_α=-17.3		(-61, -40, 176)	(-63, -38, 183)
		(-10, -8, 191) (1, -8, 185)	V_α =-11.1			
	flute	(-58, 6, 205)			(-117, -40, 170)	(-123, -39, 183)
		(-63, -2, 198) (-53, -3, 198)				
6	violin	(45, 1, 198) (56, 1, 198)			(-64, -47, 185)	(-63, -38, 183)
		(41, -2, 191) (52, -13, 191)				
	flute	(-3, 4, 205)	H_α=-31.9	H_α=-40.7	(-116, -45, 185)	(-123, -39, 183)
		(-10, -4, 213) (0, -5, 205)	V_α =-11.4			

10 Observations

In this paper, we have described our experiments on feature extraction and integration of vision and audition using a map. We would like to make the following observations:

1. Since the map consists of only a symbolic representation with spatial and temporal relationships of objects, a *map is updated only when another event occurs* (like motion or a new sound source). This framework saves a lot of computational time and is not restricted to a limited number of objects.
2. In the experiments of this paper, a *visual and a sound object are associated* with each other via a simple, a-priori known, knowledge base. That is, the shape of an object specifies the pitch of a sound and vice versa. For a more complicated object association, a knowledge base can be augmented by using, e.g., sound ontology [31].
3. The *visual field angles* of the camera we use in the experiments is *limited* to about $45^\circ \times 34^\circ$. Even if we use a wider angle, despite of the drop in spatial resolution, it will never reach the full 360 degrees. Therefore, other perceptual input, sound in particular, is needed.
4. *Occluded images* can be solved in the same manner as if the object was outside the visual field. With vision only, additional visual features such as color are needed to facilitate object recognition. Sound hardly changes when objects are (partially) occluded, hence recognition by sound, in such case, is much easier.
5. *Moving objects* are difficult to detect by sounds only, because the ambiguity in the localization of a sound source is about $\pm 10^\circ$. Moving objects can be localized by vision far more accurately, which can be used in turn to improve the sound source separation.
6. The current implementation of the event generator used for active vision and audition is very simple and can be improved considerably. Consequently, by creating the scenario, we considered these limitations.

11 Conclusions and Future Research

We have presented promising results of a selective attention model in a simple attention environment. We mainly focussed on an application where both vision and audition benefit from each other. The input of the model is formed by specific and robust features that are extracted from a huge amount of sensor data. Seven different types of features are used; *edges*, *corners*, *stereo* to create a 3D coordinate system for the detected edges and corners. *Speed* and *direction* are obtained in temporal space by deriving change in position at different timestamps. For sound *pitch* and *direction* are extracted.

In practice the attention model will perform well if all objects are recognized, i.e. in the current implementation all corners should be detected. Recognition fails if a corner of an object is occluded. When all objects are recognized, selective attention is a relatively simple task. The results of the model in the used simulation showed accurate behavior compared to the measured data. Due to limited resolution there were some small differences.

Audition was employed to improve selective attention because the visual fields of the cameras are small or objects are (partially) occluded. Sound and vision benefit from each other when attention is triggered by sound. In the scenario an illustration of this symbiosis is given by playing the flute, which is initially outside the visual fields of the cameras. It is known that humans highly rely on visual cues in acoustically noisy and ambigious environments [7]. Experimental psychologists have long been studying the influence of visual factors in human auditory spatial perception. Experiments revealed that people wearing 180 degree rotating prismatic glasses for several days localized sounds opposed to their real physical location. This gives strong evidence that sound is strongly attached to a visual object [37]. Other experiments have shown that the accuracy of auditory localization is increased if subjects can focus on the sound source [16]. Interaction between audition and vision that is beneficial for both cues is still found rarely in literature.

In the model, a map is used to integrate location of both vision and audition cues. This map is needed to store all recognized objects in the local scene even if they are currently outside the visual field. It also facilitates and speeds up the selective attention model. A learning mechanism is incorporated as well, to make the model adaptive to any arbitrary scene. In the current paper we assumed that learning was already accomplished, but in future research we will focus on adaptive learning mechanisms too.

Working towards and intelligent perception model there are still many aspects that can be improved or extended. For example, other sound source features like *timbre*, which is used for identification [17], can be included as well. Also edge extraction can be optimized and color can be incorporated in the model. Further development of an intelligent matching algorithm to solve stereo vision in a general way. Further the event generator can be improved considerably.

Finally general mechanisms for learning and knowledge management, i.e. how to keep the knowledge database compact and retrieve knowledge in a fast and easy way will be elaborated on to finally create a humanoid that simulates intelligent behavior.

References

1. D. H. Ballard and C. M. Brown. *Computer Vision*. Prentence-Hall Inc., 1982.
2. S. F. Boll. A spectral subtraction algorithm for suppression of acoustic noise in speech. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP-79)*, pages 200–203, 1979.
3. H. Bunke and B. T. Messmer. Efficient attributed graph matching and its application to image analysis. In C. Braccini, L. DeFloriani, and G. Vernazza, editors, *8-th International Conf. ICIAP'95, Lecture Notes in Computer Science -Image Analysis and Processing*, volume 974, pages 45–55, San Remo, Italy, September 1995.
4. J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(9):679–698, November 1986.
5. S. Cavaco and J. Hallam. A biologically plausible acoustic azimuth estimation system. In *Proceedings of IJCAI-99 Workshop on Computational Auditory Scene Analysis (CASA'99)*, pages 78–87, Stockholm, Sweden, August 1999.
6. C. Dillon and T. Caelli. Learning image annotations: The CITE system. *Videre: Journal of Computer Vision Research*, 1(2):90–121, Winter 1998.
7. N. Erber. Auditory-visual perception of speech. *Journal of speech and hearing disorders*, pages 481–492, 1975.
8. M. A. Eshera and King-Sun Fu. An image understanding system using attributed symbolic representation and inexact graph-matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(5):604–618, September 1986.
9. J. P. Gottlieb, M. Kusunoki, and M.E. Goldberg. The representation of visual saliency in monkey parietal cortex. *Nature*, 391(6666):481–484, January 1998.
10. R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*. Addison-Wesley, 1992.
11. F. Heitger, L. Rosenthaler, R. von der Heydt, E. Peterhans, and O. Kübler. Simulation of neural contour mechanisms: from simple to end-stopped cells. *Vision Research*, 32(5):963–981, 1992.

12. D. Hubel and T. Wiesel. Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex. *Journal of Physiology, London*, 160:106–154, 1962.
13. D. H. Hubel and T. N. Wiesel. Integrative action in the cat's lateral geniculate body. *Journal of Physiology, London*, 155:385–398, 1961.
14. D. H. Hubel and T. N. Wiesel. Receptive fields of cells in striate cortex of very young visually inexperienced kittens. *Journal of Neurophysiology*, 26:994–1002, 1963.
15. L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, November 1998.
16. B. Jones and B. Kabanoff. Eye movements in auditory space perception. *Perception and Psychophysics*, 17(3):241–245, 1975.
17. K. Kashino and H. Murase. Sound source identification for ensemble music based on the music stream extraction. In *Working Notes of the Computational Auditory Scene Analysis Workshop (IJCAI-97)*, pages 127–134, August 1997.
18. H. Kitano, H. G. Okuno, K. Nakadai, I. Fermin, T. Sabisch, Y. Nakagawa, and T. Matsui. Designing a humanoid head for robocup challenge. In *Agent 2000*, 2000.
19. C. Koch and S. Ullman. Shifts in selective visual attention: Towards the underlying neural circuitry. *Human Neurobiology*, 4:219–227, 1985.
20. P. Kovesi. Image features from phase congruency. *Videre: Journal of Computer Vision Research*, 1(3):2–26, 1999.
21. T. P. Lindeberg. *Scale Space Theory in Computer Vision*. Kluwer, Dordrecht, The Netherlands, 1994.
22. T. Lourens. *A Biologically Plausible Model for Corner-based Object Recognition from Color Images*. Shaker Publishing B.V., Maastricht, The Netherlands, March 1998.
23. T. Lourens and R. P. Würtz. Object recognition by matching symbolic edge graphs. In R. Chin and T. C. Pong, editors, *Proceedings of the Third Asian Conference on Computer Vision, ACCV '98*, volume 1352 of *Lecture Notes in Computer Science*, pages 193–200. Springer-Verlag, January 1998.
24. T. Lourens and R. P. Würtz. Extraction and matching of symbolic contour graphs. Submitted, 2000.
25. B. T. Messmer and H. Bunke. A new algorithm for error-tolerant subgraph isomorphism detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):493–504, 1998.
26. R. Mohr. Projective geometry and computer vision. In C. H. Chen, L. F. Pau, and P. S. P. Wang, editors, *Handbook of Pattern Recognition and Computer Vision*, pages 369–393, Singapore, 1993. World Scientific.
27. M. Morrone, J. Ross, D. Burr, and R. Owens. Mach bands are phase dependent. *Nature*, 324(6094):250–253, 1986.
28. M. C. Morrone and D. C. Burr. Feature detection in human vision: A phase-dependent energy model. *Proc. of the Royal Society of London*, 235:335–354, 1988.
29. K. Nakadai, T. Lourens, H. G. Okuno, and H. Kitano. Active audition for humanoid. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 832–839, 2000.
30. Y. Nakagawa, H. G. Okuno, and H. Kitano. Using vision to improve sound source separation. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pages 768–775. AAAI Press/The MIT Press, 1999.
31. T. Nakatani and H. G. Okuno. Sound ontology for computational auditory scene analysis. In *Proceedings of 15th National Conference on Artificial Intelligence (AAAI-98)*, pages 1004–1010. AAAI, 1998.
32. T. Nakatani and H. G. Okuno. Harmonic sound stream segraegation using localization and its application to speech stream segregation. *Speech Communication*, 27(3-4):209–222, April 1999.
33. E. Niebur and C. Koch. Computational architectures for attention. In R. Parasuraman, editor, *The Attentive Brain*, pages 163–186. MIT Press, Cambridge, Mass., 1998.
34. D. L. Robinson and S. E. Peterson. The pulvinar and visual salience. *Trends in Neurosciences*, 15(4):127–132, April 1992.
35. P. L. Rosin. Edges: Saliency measures and authomatic thresholding. Technical Report I.95.58, Institute for Remote Sensing Applications, Ispra, Italy, May 1995.
36. M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. PWS Publishing, 1999.
37. G. Stratton. Vision without inversion of the retinal image. *Psychol. Rev.*, pages 341–389, 463–481, 1897.
38. A. M Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12(1):97–136, January 1980.
39. J. K. Tsotsos, S. M. Culhane, W. Y. K. Wai, Y. K. Lai, N. Davis, and F. Nuflo. Modelling visual attention via selective tuning. *Artificial Intelligence*, 78(1-2):507–545, October 1995.
40. T. Ueshiba, Y. Kawai, Y. Ishiyama, Y. Sumi, and F. Tomita. An efficient matching algorithm for segment-based stereo vision using dynamic programming technique. In *Proceedings of IAPR Workshop on Machine Vision Applications*, pages 61–64, Makuhari, Japan, November 1998.
41. J. R. Ullman. An algorithm for subgraph isomorphism. *Journal of the Association for Computing Machinery*, 23(1):31–42, 1976.
42. S. Venkatesh and R. Owens. On the classification of image features. *Pattern Recognition Letters*, 11:339–349, 1990.
43. S. Venkatesh and P. L. Rosin. Dynamic threshold determination by local and globall edge evaluation. *Computer Vision, Graphics and Image Processing*, 57(2):146–160, 1995.
44. R. P. Würtz and T. Lourens. Corner detection in color images by multiscale combination of end-stopped cortical cells. In W. Gerstner, A. Germond, M. Hasler, and J. D. Nicoud, editors, *Proceedings of the International Conference on Artificial Neural Networks, ICANN'97*, volume 1327 of *Lecture Notes in Computer Science*, pages 901–906. Springer Verlag, October 1997.
45. R. P. Würtz and T. Lourens. Corner detection in color images through a multiscale combination of end-stopped cortical cells. *Image and Vision Computing*, 18(6-7):531–541, April 2000.