Distributed Visual Attention on a Humanoid Robot

Aleš Ude[1,2,3] Valentin Wyart[2] Li-Heng Lin[2] Gordon Cheng[1,2]

[1]*Japan Science and Technology Agency*
ICORP Computational Brain Project
2-2-2 Hikaridai, Seika-cho
Soraku-gun, Kyoto, Japan
{aude,gordon}@atr.jp

[2]*ATR Computational Neuroscience Lab.*
Dept. of Humanoid Robotics and Comput.
Neurosc., 2-2-2 Hikaridai, Seika-cho
Soraku-gun, Kyoto, Japan
valentin@atr.jp

[3]*Jozef Stefan Institute, Dept. of*
Automatics, Biocybernetics, and
Robotics
Jamova 39, 1000 Ljubljana, Slovenia

Abstract— **Complex visual processes such as visual attention are often computationally too expensive to allow real-time implementation on a single computer. To solve this problem we study distributed computer architectures that enable us to divide complex tasks into several smaller problems. In this paper we demonstrate how to implement distributed visual attention system on a humanoid robot to achieve real-time operation at relatively high resolutions and frame rates. We start from a popular theory of bottom-up visual attention that assumes that information across various modalities is used for the early encoding of visual information. Our system uses five different modalities including color, intensity, edges, stereo, and motion. We show how to distribute the attention processing on a computer cluster and study the issues arising on such systems. The system was fully implemented on a workstation cluster comprised of eight PCs. It was used to drive the gaze of a humanoid head towards potential regions of interest.**

Index Terms— **Visual attention, distributed computing, humanoid heads.**

I. Introduction

There are converging lines of evidence for the existence of separate visual areas in the brain, each specialized for processing a particular aspect of the visual scene [1]. One possible explanation for such an architecture was given by David Marr [2]:

> Any large computation should be split up and implemented as a collection of small sub-parts that are as nearly independent of one another as the overall task allows. If a process is not designed in this way, a small change in one place will have consequences in many other places. This means that the process as a whole becomes extremely difficult to debug or to improve, because a small change to improve one part has to be accompanied by many simultaneous compensating changes elsewhere.

We believe that distributed processing of visual information is the key to the real-time operation of complex visual processes in technical systems such as humanoids. In this paper we propose a distributed implementation of a visual attention system on a humanoid robot. Visual attention is a complex, but comparatively well understood process in psychology and computational neuroscience. It is therefore well suited as an experimental testbed for distributed processing of visual information.

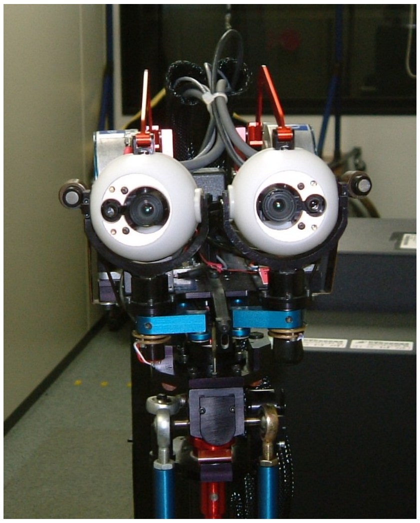

Fig. 1. Humanoid head used in the experiments. It has a foveated vision system and 7 degrees of freedom (two DOFs in each eye and three DOFs in the neck).

Among several biologically plausible models for visual attention, approaches that build on feature integration theory [3] and saliency maps [4] resulted in most useful architectures [5]–[7], including some implementations on humanoid robots [8]–[11]. We started from the saliency map model proposed in [6]. This model exhibits a distributed processing architecture which is quite typical for the processing of information in the brain. The original visual stream is subdivided into several processing streams that are in part independent of each other. The processing time and consequently the latency and frame rates can vary across the streams. Further down the processing stream the results must be integrated and synchronized to produce the global saliency map suitable for the selection of the focus of attention.

It is evident from various visual disabilities that the ability of the brain to reassign the processing of visual information to new areas is rather limited and that it also takes time. Instead visual information is transferred along a number of pathways (e. g. magnocellular pathway, parvocellular-blob pathway, and parvocellular-interblob pathway [12]) and visual processes are executed in well defined areas of the brain. Visual perception results from interconnections between these partly separate and functionally specialized systems. This was the guiding

0-7803-9320-1/05/$20.00 ©2005 IEEE

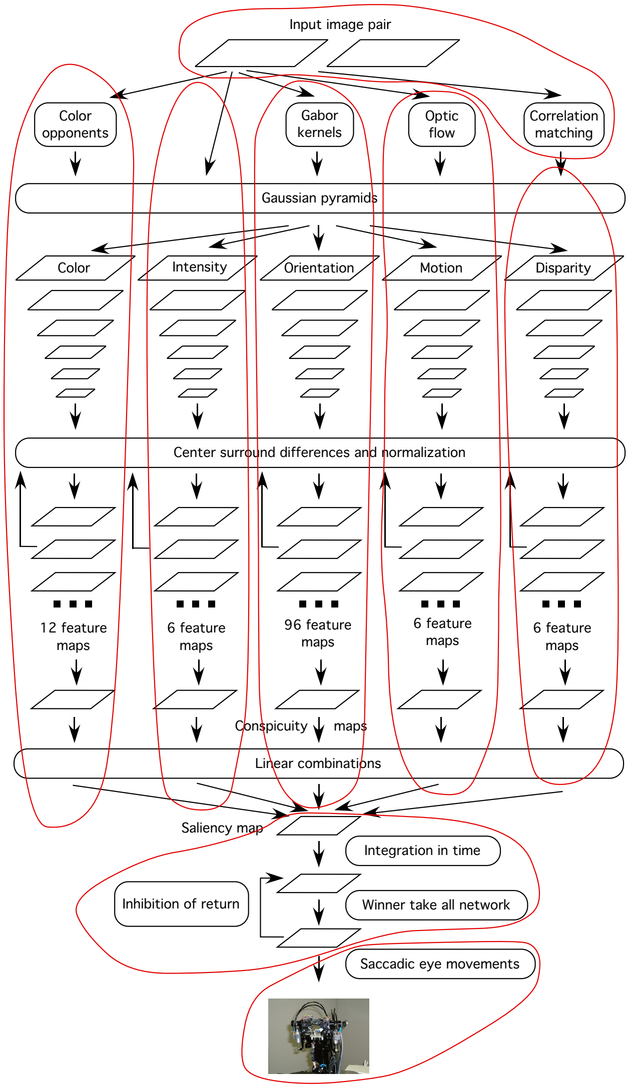

Fig. 2. Visual attention architecture. Besides the robot and saccadic movements, there are two additional streams compared to the architecture proposed in Ref. [6]: motion and disparity. They are both associated with the magnocellular processing pathway, whereas color, intensity, and orientation belong to the parvocellular pathway. In addition, our system applies Gabor kernels at different scales, not just at different orientations. The distribution of visual processes across the computer cluster is indicated by red circles. Each circle encloses the processes executed by one computer.

principle for the design of our attention system. Our goal was to define a system that will allow us to transfer information from the source to a number of computers executing specialized vision processes, either sequentially or in parallel, and to provide means to integrate information from various streams coming at different frame rates and with different latencies. The transfer of information can be both feed-forward (bottom-up processing) and feed-backward (top-down effects).

II. Bottom-up attention system

Feature integration theory of attention postulates that bottom-up preattentive processing is based on exploring the visual search space for various features and integrating them, e. g. by way of saliency maps, until the location of the most salient area in the image emerges, e. g. through the competition across various feature maps. The generation of a number of feature maps at full resolution and at high frame rates is a computationally intensive process. Unless we are ready to make compromises about the image resolution and/or frame rate, we need to utilize a distributed computer architecture.

An attention system based on saliency maps decomposes the visual input into several streams, each of them corresponding to one type of retinal feature maps calculated at different scales (Section II-A). Within each feature processor, maps at different scales are combined to generate a global conspicuity map that emphasizes locations that stand out from their surrounding (Section II-B). The conspicuity maps are combined into a global saliency map, which encodes the saliency of image locations over the entire feature set (Section II-C). The time-integrated global saliency map is used as an input to a winner-take-all neural network, which is used to compute the most salient area in the image stream (Section II-D). This architecture is presented in Fig. 2. While the described approach is purely bottom-up, top-down effects can be introduced by biasing the weights when combining the conspicuity maps or by introducing lateral inhibition when computing local feature maps.

A. Generation of early feature maps

The current version of the system includes the processing of color, intensity, orientation, motion, and disparity (see Fig. 2). It would be impossible to implement all these processes on one computer and maintain the frame rate and resolution. Among the implemented feature processors, the generation of orientation maps is the most computationally intensive one. We followed the biologically motivated implementation suggested in [6], where orientation feature maps were computed by applying Gabor kernels

$$\Phi(\boldsymbol{x}) = \frac{\|\boldsymbol{k}_{\mu,\nu}\|^2}{\sigma^2}\cdot\exp\left(-\frac{\|\boldsymbol{k}_{\mu,\nu}\|^2\|\boldsymbol{x}\|^2}{2\sigma^2}\right)\cdot \left(\exp\left(i\boldsymbol{k}_{\mu,\nu}^T\boldsymbol{x}\right)-\exp\left(-\frac{\sigma^2}{2}\right)\right), \tag{1}$$

where $\boldsymbol{k}_{\mu,\nu} = k_\nu[\cos(\phi_\mu),\ sin(\phi_\mu)]^T$. In Ref. [6] only one scale k_ν and 4 orientations $\phi_\mu = 0,\,45,\,90,\,135$ were used. Gabor kernels were suggested to model the receptive fields of simple cells in primary visual cortex. It has been shown, however, that there exist simple cells sensitive not only to specific positions and orientations, but also to specific scales. We therefore utilized not only Gabor kernels at 4 different orientations but also at 4 different scales.

Color and intensity were dealt with in a similar way as in [6]. For the calculation of motion, we used a variant of Lucas-Kanade algorithm. A correlation-based technique was used to generate disparity maps at 30 Hz.

B. Center-surround differences

At full resolution (320×240), the above feature processors generate 2 early feature maps for color (based on double color opponents theory [1]), 1 for intensity, 16 for orientation, 1

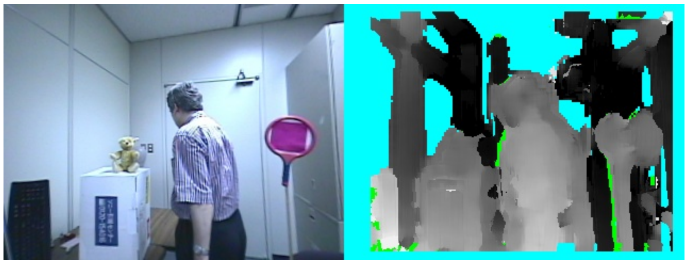

Fig. 3. The captured image and the associated disparity map. This data is sent from the frame grabber to all feature processors (compare with Fig. 2). The green and cyan parts of disparity maps show the areas where the disparities are not available.

for motion and 1 for disparity. Center-surround differences were suggested as a computational tool to detect local spatial discontinuities in feature maps which stand out from their surround [6]. Center-surround differences can be computed by first creating Gaussian pyramids out of the 21 early feature maps. From the uppermost scale $\boldsymbol{I}_{\mathrm{f}}(0)$, where f is the corresponding feature, maps at lower scales are calculated by filtering of the map at the previous scale with a Gaussian filter. The resolution of the map at lower scale is half the resolution of the map at the scale above it. The creation of Gaussian pyramids is straightforward for all of the above features except for disparities. Disparity maps are never fully populated (see Fig. 3) and the missing data is labeled as such in the resulting pyramids.

Center-surround differences are calculated by subtracting pyramids at coarser scale from the pyramids at finer scale. These differences can be calculated by up-sampling the pyramids at coarser scales to finer scales. We calculated the center-surround differences between the pyramids $\boldsymbol{I}_{\mathrm{f}}(c)$, $c = 2, 3, 4$ and $\boldsymbol{I}_{\mathrm{f}}(s)$, $s = c + \delta$, $\delta = 2, 3$. This results in 6 maps per feature. The calculation of disparity feature maps is a bit more complicated because we need to take make sure that the missing data is ignored. The center-surround differences involving missing data are hence set to zero.

C. Saliency maps

The combination of feature maps into conspicuity maps for color $\boldsymbol{I}_{\mathrm{c}}$, intensity $\boldsymbol{I}_{\mathrm{b}}$, orientation $\boldsymbol{I}_{\mathrm{o}}$, motion $\boldsymbol{I}_{\mathrm{m}}$, and disparities $\boldsymbol{I}_{\mathrm{d}}$ involves normalization to a fixed range and searching for global and local maxima to promote feature maps with strong global maxima. For each modality, feature maps are combined at the coarsest scale. This process is equivalent to [6] and we omit the details here. The conspicuity maps are finally combined into a global saliency map

$$S = w_{\mathrm{c}}\boldsymbol{I}_{\mathrm{c}} + w_{\mathrm{b}}\boldsymbol{I}_{\mathrm{b}} + w_{\mathrm{o}}\boldsymbol{I}_{\mathrm{o}} + w_{\mathrm{m}}\boldsymbol{I}_{\mathrm{m}} + w_{\mathrm{d}}\boldsymbol{I}_{\mathrm{d}}. \tag{2}$$

The weights w_{c}, w_{b}, w_{o}, w_{m}, and w_{d} can be set based on top-down information about the importance of each modality. In the absence of top-down information, they can be set to a fixed value, e. g. $\frac{1}{5}$.

On a humanoid robot we are interested in processing time-varying visual information. Since we use standard NTSC cameras on our humanoid head (see Fig. 1), the saliency maps can change up to 30 times per second. We therefore integrate the information in time to provide better input to the competitive process that selects the focus of attention

$$S_{\mathrm{int}}(t) = \gamma^{\delta} S_{\mathrm{int}}(t - \delta) + G_{\sigma} * S(t),\ 0 < \gamma < 1, \tag{3}$$

where $\delta \geq 1$ is the difference in the frame index from the previous saliency map and $G_{\sigma} * S(t)$ is the convolution of the current saliency map with the Gaussian filter with standard deviation σ.

D. Winner-take-all network and inhibition of return

The aim of the preattentive mode of visual processing is to select the focus of attention, which is subsequently processed more exhaustively to solve higher-level tasks such as object recognition [4]. Although there exist attempts to construct a more integrated processing model that encompasses both preattentive processing and object recognition [13], it is still not clear that such models will be able to overcome the combinatorial explosion associated with the analysis of shape and recognition across the whole visual scene. Two-stage processing models therefore still seem to be the best choice for implementation on a technical system.

Winner-take-all network has been suggested as means to calculate the focus of attention from the saliency map [4]. We used the leaky integrate-and-fire model to build a three layer 2-D neural network of first order integrators to integrate the contents of the saliency map and choose a focus of attention over time. It is based on the integration of the following system of differential equations:

$$u_1(\mathbf{x}, t) = \sum_{\mathbf{y}} w_1(\mathbf{x}, \mathbf{y}) S_{\mathrm{int}}(\mathbf{y}, t), \tag{4}$$

$$\begin{aligned} \frac{\mathrm{d}u_2}{\mathrm{d}t}(\mathbf{x}, t) + \frac{1}{\tau_2} u_2(\mathbf{x}, t) &= \sum_{\mathbf{y}} w_2(\mathbf{x}, \mathbf{y}) u_1(\mathbf{y}, t) - \\ &\quad \sum_{\mathbf{y}} w_{\mathrm{co}}(\mathbf{x}, \mathbf{y}) u_3(\mathbf{y}, t), \quad (5) \\ \frac{\mathrm{d}u_3}{\mathrm{d}t}(\mathbf{x}, t) + \frac{1}{\tau_3} u_3(\mathbf{x}, t) &= \sum_{\mathbf{y}} w_3(\mathbf{x}, \mathbf{y}) u_2(\mathbf{y}, t), \quad (6) \end{aligned}$$

where $u_i(\mathbf{x}, t)$ is the membrane potential of the neuron of the i-th layer located at $\mathbf{x} = (x, y)$ at time t, τ_i is the time constant of the i-th layer, $w_i(\mathbf{x}, \mathbf{y})$ is the weighting function of input synaptic connections of the i-th layer between locations $\mathbf{x}$ and $\mathbf{y}$ and $w_{\mathrm{co}}(\mathbf{x}, \mathbf{y})$ is the weighting function of synaptic connections of the second layer. Function $w_i(\mathbf{x}, \mathbf{y})$ models spatial convergence between successive layers and is given by:

$$w_1(\mathbf{x}, \mathbf{y}) = w_2(\mathbf{x}, \mathbf{y}) = w_3(\mathbf{x}, \mathbf{y}) = \frac{1}{2\pi\sigma_{\mathrm{in}}^2} \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma_{\mathrm{in}}^2}\right) \tag{7}$$

Functions $w_{\mathrm{co}}(\mathbf{x}, \mathbf{y})$ model the coupling effects between the neurons of the network including long-range inhibition and

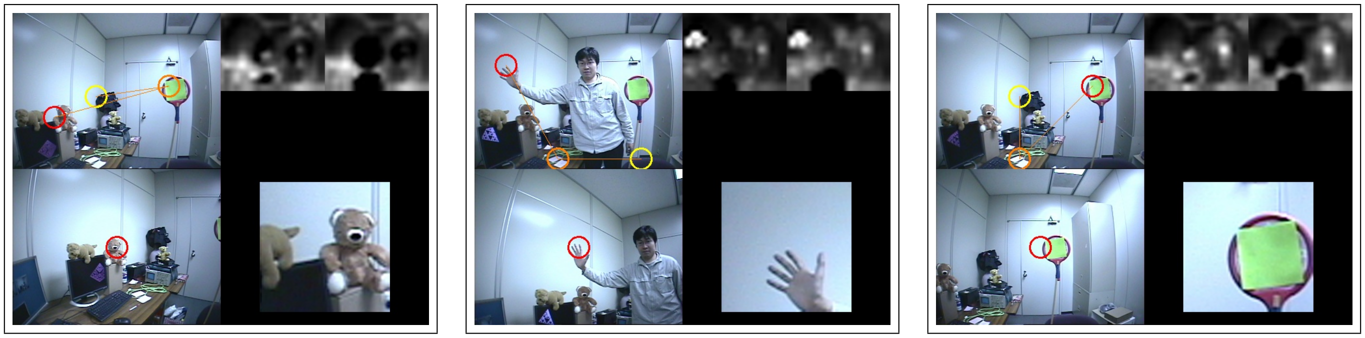

Fig. 4. Focus of attention. In each of the three images, the left upper corner shows the last image used in preattentive processing with the three last most salient locations indicated by red, orange, and yellow circle. The left lower corner shows the view from the peripheral camera after the saccade. The two smaller images in the upper right corner show the first two layers (u_1 and u_2) from the system of differential equations (4) - (6). The third layer has already been reset because the images show the situation after the eyes saccaded towards the position indicated by the winner neuron. Finally, the image in the lower right corner shows the foveal view of the area of interest which could be used in postattentive processing. Note that the system detected areas that are intuitively salient.

short-range excitation to produce the winning neuron and is given by:

$$
\begin{aligned}
w_{\text{co}}(\mathbf{x},\mathbf{y}) \quad = \quad & \frac{c_{\text{inh}}^2}{2\pi\sigma_{\text{inh}}^2}\exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma_{\text{inh}}^2}) - \\
& \frac{c_{\text{exc}}^2}{2\pi\sigma_{\text{exc}}^2}\exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma_{\text{exc}}^2}) \qquad (8)
\end{aligned}
$$

We used Euler's method to integrate equations (5) and (6). The integration frequency was 100 Hz and is different from the timing of the vision signal (30 Hz). Hence before updating the integrated saliency map S_{int}, equations (5) and (6) are integrated a few times as temporal smoothing. When the membrane voltage of one of the neurons of the third layer $u_3(\mathbf{x}, t)$ reaches an adaptive firing threshold $u_{\text{fire}}(t)$, the robot moves its eyes so that the most salient location is placed over the fovea. Vision processing is suppressed during the saccade. Since postattentive processing has not been integrated into the current version of the system yet, the robot just waits for 500 ms before moving its eyes back to the original position. At this point the neurons of the third layer are reset to their ground membrane voltage as global lateral inhibition and a local inhibitory signal is smoothly propagated from the first to the second layer at the attended location as inhibition of return. The strength of this inhibitory effect is gradually reduced as the time passes to allow for further exploration of the previously attended regions.

III. Top-down effects

As already mentioned in Section II-C, top-down effects can be introduced into the system by selecting different weights when combining the conspicuity maps into a single saliency map by means of Eq. (2). This is, however, still very general and does not allow for the promotion of specific features such as for example a red target. Just increasing the weight of a color-based conspicuity map would promote any color, not just the red one. One way to accomplish this has been proposed in FeatureGate model of human visual attention [8]. This model suggests to introduce top-down effects by lateral inhibition of activation in feature maps. The lateral inhibition is activated if there exists an area in the neighborhood that differs from the top-down feature $\mathrm{f_t}$ less than the current location $\boldsymbol{x}$. We note here that it has been suggested already in [4] that early feature maps are the proper place to implement lateral inhibitory effects.

Let $\mathcal{N}_c(\boldsymbol{x})$ be the neighborhood of location $\boldsymbol{x}$ at level c in the pyramid and let $\mathcal{S}_c(\boldsymbol{x})$ be all pixels in the neighborhood that are closer to the target than $\boldsymbol{x}$

$$
\mathcal{S}_c(\boldsymbol{x}) = \{\boldsymbol{y} \in \mathcal{N}_c(\boldsymbol{x});\ \rho(\boldsymbol{I}(\boldsymbol{y}), \mathrm{f_t}) < \rho(\boldsymbol{I}(\boldsymbol{x}), \mathrm{f_t})\} \qquad (9)
$$

The activation in the early feature map is decreased by the value proportional to the difference in the distance from the target feature

$$
\boldsymbol{I}'_{\mathrm{f}}(c,\boldsymbol{x}) = \boldsymbol{I}_{\mathrm{f}}(c,\boldsymbol{x}) - \beta \sum_{\boldsymbol{y}\in\mathcal{S}_c(\boldsymbol{x})} \{\rho(\boldsymbol{I}(\boldsymbol{x}), \mathrm{f_t}) - \rho(\boldsymbol{I}(\boldsymbol{y}), \mathrm{f_t})\} \qquad (10)
$$

It is task dependent if all or only some of the early feature maps $\boldsymbol{I}_{\mathrm{f}}$ needs to take into account guidance provided by the top-down feature $\mathrm{f_t}$.

Although the integration of top-down effects into the distributed visual attention system has not been completed yet, it is clear that the architecture allows for such an addition; each top-down effect can be implemented on one computer and the resulting data can be broadcast to the early feature map generators that need to incorporate top-down cues.

IV. Saccadic eye movements

Directing a spotlight of attention towards interesting areas involves saccadic eye movements. It is sufficient to use only the eye degrees of freedom to place the most salient area in the image stream over the fovea. The system is calibrated and we can easily calculate the pan and tilt angle for each eye that are necessary to direct the gaze towards the desired location. Human saccadic eye movements are very fast, thus the current version of our eye control system simply moves the robot eyes towards the desired configuration as fast as possible. Three examples are shown in Fig. 4, where the red circled areas represent the focus of attention before and after the saccade. The accuracy of control was increased by processing rectified images (see Fig. 3), in which distortion effects are corrected.

TABLE I

Frame indexes of simultaneously processed images under different synchronization schemes. In each box, ordered from left to right column, the frame indices belong to the disparity, color, orientation, intensity, and motion conspicuity map. See text in Section V-A for further explanation.

70656	70656	70656	70656	70656	61250	61249	61250	61250	61250	65911	65910	65907	65911	65912
70675	70675	70675	70675	70675	61251	61251	61250	61251	61251	65912	65910	65910	65911	65913
70678	70678	70678	70678	70678	61252	61251	61250	61251	61252	65912	65912	65910	65912	65914
70695	70695	70695	70695	70695	61253	61253	61250	61253	61253	65913	65912	65910	65913	65915
70711	70711	70711	70711	70711	61253	61253	61254	61254	61254	65914	65912	65910	65913	65916
70715	70715	70715	70715	70715	61255	61253	61254	61254	61255	65915	65914	65913	65915	65917
70724	70724	70724	70724	70724	61256	61256	61254	61256	61256	65917	65914	65913	65916	65918
70757	70757	70757	70757	70757	61257	61256	61257	61257	61257	65918	65916	65913	65916	65919
70758	70758	70758	70758	70758	61258	61258	61257	61257	61258	65918	65916	65916	65918	65920
70777	70777	70777	70777	70777	61259	61258	61257	61259	61259	65919	65918	65916	65919	65921
70790	70790	70790	70790	70790	61260	61260	61260	61260	61260	65920	65918	65916	65921	65922
70799	70799	70799	70799	70799	61260	61260	61260	61261	61261	65921	65921	65919	65922	65923
70802	70802	70802	70802	70802	61262	61262	61260	61261	61262	65923	65921	65919	65922	65924
70815	70815	70815	70815	70815	61263	61262	61260	61263	61263	65924	65923	65919	65923	65925
70837	70837	70837	70837	70837	61264	61264	61264	61264	61264	65925	65923	65922	65923	65926

V. Distributed processing of visual streams

The distributed architecture presented in Fig. 2 is essential to achieve real-time operation of the complete visual attention system. In the current implementation, all of the computers are connected to a single switch via a gigabit Ethernet. We use UDP protocol for data transfer. Data that needs to be transferred from the image capture PC includes the rectified color images captured by the left camera, which are broadcast from the frame grabber to all other computers on the network, and the disparity maps, which are sent directly to the PC that takes care of the disparity map processing. Full resolution (320×240 to avoid interlacing effects) was used when transferring and processing these images. The five feature processors send the resulting conspicuity maps to the PC that deals with the calculation of the saliency maps and with the integration of the winner-take-all network. Finally, the position of the most salient area in the image stream is sent to the PC taking care of motor control. The current setup with all the computers connected to a single gigabit switch proved to be sufficient to transfer the data at full resolutions and frame rates. However, our implementation of the data transfer routines allows us to split the network into a number of separate networks should the data load become too large. This is essential to make it possible to scale the system to a more advanced vision processing such as shape analysis and object recognition.

Of the 8 workstations on the network, 4 run Windows 2000, 3 Windows XP and 1 Linux. Five of the PCs are equipped with 2×2.2 GHz Intel Xeon processors, two with 2×2.8 GHz Intel Xeon processors, and one with 2 Opteron 250 processors. Such a heterogeneous computer cluster in which every computer needs to solve a different problem will necessarily result in visual streams coming at different frame rates and with different latencies. In the following we describe how to ensure smooth operation under such conditions.

A. Synchronization

The processor that needs to solve the most difficult synchronization task is the one that integrates the conspicuity maps into a single saliency map. It receives input from five different feature processors. The slowest among them is the orientation processor that can take care of only every third frame. On the other hand, the disparity processor works at full frame rate and with lower latency because the data it needs is sent directly to this processor instead of being broadcast across the network. While it would be possible to further distribute the processing load of the orientation processor, we did not follow this approach because our computational resources are not unlimited. We were much more interested in designing a general synchronization scheme that would allow us to realize real-time processing under such conditions.

The simplest approach to synchronization is to ignore the different frame rates and latencies and to simply process the data that was last received from each of the feature processors. Some of the resulting frame indices for conspicuity maps that are in this case combined into a single saliency map are shown in the rightmost box of Tab. I. Note that the time difference (frame index) between simultaneously processed conspicuity maps is quite large, up to 6 frames (or 200 milliseconds). It does not happen at all that conspicuity maps with the same frame index would be processed simultaneously.

Ideally, we would always process only data captured at the same moment in time. This, however, proves to be impractical when integrating the five conspicuity maps. To achieve full synchronization, we associated a buffer with each of the incoming data streams. The integrating process received the requested conspicuity maps only if data from all five streams was simultaneously available. The results are shown in the leftmost box of Tab. I. Note that lots of data is lost when using this synchronization scheme because images from all five processing streams are only rarely available.

We have therefore implemented a scheme that represents a compromise between the two approaches. Instead of requesting that the data is fully synchronized, we monitor the buffer and simultaneously process the data that is as close together in time as possible. This can be accomplished by waiting that for each data stream, there is data available with the time

```
request for data with frame index $n$:
    get access to buffers and lock writing
    $r = 0$
    for $i = 1, \ldots, m$
        find the smallest $b_{i,j}$ so that $n < b_{i,j}$
        if such $b_{i,j}$ does not exist
            reply images with frame index $n$ not yet available
            unlock buffers and exit
        if $b_{i,(j-1)\%M} \leq n$
            $j_i = b_{i,(j-1)\%M}$
        else
            $r = \max(r, b_{i.j})$
    if $r > 0$
        reply $r$ is the smallest currently available frame index
        unlock buffers and exit
    return $\{\boldsymbol{I}_{1,j_1}, \ldots, \boldsymbol{I}_{m,j_m}\}$
    unlock buffers and exit
```

Fig. 5. Pseudo-code for the delayed synchronization algorithm. m denotes the number of incoming data streams, or – in other words – the number of preceding nodes in the network of visual processes. To enable synchronization of data streams coming with variable latencies and frame rates, each data packet (image, disparity map, conspicuity map, join angle configuration, etc.) is written in the buffer associated with the data stream, which has space for M latest packets. $b_{i,j}$ denotes the frame index of the j-th data packet in the buffer of the i-th processing stream. $\boldsymbol{I}_{i,j}$ are the data packets in the buffers and m is the number of data streams coming from previous processes.

stamp before (or at) the requested time as well as data with the time stamp after the requested time. This allows us to optimally match the available data. The algorithm is given in Fig. 5. For this delayed synchronization scheme, the frame indices of simultaneously processed data are shown in the middle box of Tab. I. It is evident that all of the available data is processed and that frames would be skipped only if the integrating process is slower than the incoming data streams. The time difference between the simultaneously processed data is cut to half (3 frames or 100 milliseconds). However, the delayed synchronization scheme does not come for free; since we need to wait that at least two frames from each of the data streams are available, the latency of the system is increased by the latency of the slowest stream. Nevertheless, we chose the delayed synchronization scheme as the method of choice for the integration of conspicuity maps.

We note here that one should be careful when selecting the proper synchronization scheme. For example, nothing less than full synchronization is acceptable if the task is to generate disparity maps from a stereo image pair. On the other hand, buffering is not desirable when the processor receives only one stream as input; it would have no effect if the processor is fast enough to process the data at full frame rate, but it would introduce an unnecessary latency in the system if the processor is too slow to interpret the data at full frame rate. The proper synchronization scheme should thus be carefully selected by the designer of the system.

VI. Summary and conclusion

Starting from the visual attention architecture proposed in [4] and [6], we designed a distributed visual attention system for a humanoid robot. We introduced several improvements to the original architecture, including the processing of orientation at different scales and the introduction of additional feature maps (motion and disparity). A system of differential equations that implements the winner-take-all network and the inhibition of return was also proposed. Finally, we suggested how to model top-down effects in early feature maps by lateral inhibition. The system was implemented on a cluster of eight workstations and used to direct the gaze of a humanoid head towards potential areas of interest. Our experimental results show that the system can select areas of interest using various features and that the selected areas are quite plausible and most of the time contain potential objects of interest (see Fig. 4).

However, the main point we wish to make in this paper is that distributed processing is necessary to achieve real-time operation of a complex vision process such as visual attention. Although some of the previous works mention that parallel implementations would be useful and indeed parallel processing was used in at least one of them [9], this is the first study that focuses on issues arising from such a distributed implementation. We developed a computer cluster architecture that allows for proper distribution of visual processes involved in visual attention. We studied various synchronization schemes that enable the integration of different processes in order to compute the final result. The designed architecture can easily scale to accommodate more complex visual processes and we intend to use it to implement further visual processes and integrate them together, thus taking a step to a more brain-like processing of visual information on humanoid robots.

References

[1] R. Sekuler and R. Blake, *Perception*, 4th ed. McGraw-Hill, 2002.

[2] D. Marr, "Early processing of visual information," *Phil. Trans. Royal Soc., Series B*, vol. 275, pp. 483–524, 1976.

[3] A. M. Treisman and G. Gelade, "A feature-integration theory of attention," *Cognitive Psychology*, vol. 12, no. 1, pp. 97–136, 1980.

[4] C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry," in *Matters of Intelligence*, L. M. Vaina, Ed. Dordrecht: D. Reidel Co., 1987, pp. 115–141.

[5] K. R. Cave, "The FeatureGate model of visual selection," *Psychological Research*, vol. 62, pp. 182–194, 1999.

[6] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, no. 11, pp. 1254–1259, 1998.

[7] J. K. Tsotsos, *Neurobiology of Attention*. Academic Press, 2005, ch. The selective tuning model for visual attention, pp. 562–569.

[8] J. A. Driscoll, R. A. Peters II, and K. R. Cave, "A visual attention network for a humanoid robot," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Victoria, Canada, 1998, pp. 1968–1974.

[9] C. Breazeal and B. Scasselatti, "A context-dependent attention system for a social robot," in *Proc. Sixteenth Int. Joint Conf. Artificial Intelligence*, Stockholm, Sweden, 1999, p. 11461151.

[10] O. Stasse, Y. Kuniyoshi, and G. Cheng, "Development of a biologically inspired real-time visual attention system," in *Biologically Motivated Computer Vision: First IEEE International Workshop*, S.-W. Lee, H. H. Bülthoff, and T. Poggio, Eds., Seoul, Korea, 2000, pp. 150–159.

[11] S. Vijayakumar, J. Conradt, T. Shibata, and S. Schaal, "Overt visual attention for a humanoid robot," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Maui, Hawaii, USA, 2001, pp. 2332–2337.

[12] E. T. Rolls and G. Deco, *Computational Neuroscience of Vision*. Oxford University Press, 2003.

[13] G. Deco and E. T. Rolls, "A neurodynamical cortical model of visual attention and invariant object recognition," *Vision Research*, vol. 44, pp. 621–642, 2004.