Câmbio: Realistic Three Dimensional Simulation of Humanoids based on Computer Vision and Robotics

CLAUS C. ARANHA[1], SCHUBERT R. CARVALHO[1], LUIZ M. G. GONÇALVEZ[2]

[1]Universidade Estadual de Campinas
Caixa Postal 6176, 13083-970, Campinas, SP, Brasil
{claus.aranha,schubert.carvalho}@ic.unicamp.br
[2]Universidade Federal do Rio Grande do Norte
DCA-Centro de Tecnologia-UFRN, 59.072-970 - Natal, RN, Brasil
lmarcos@dca.ufrn.br

Abstract. We introduce concepts and algorithms for control of visual motor commands and realistic simulation of basic abilities as visual servoing and perception for robotics simulation. The introduced tools can also be used in a straightforward way to build computer animated devices as virtual agents, and avatars. We use the tools to build part of a humanoid (arms and head) robot, effectively named "Câmbio". We will describe Câmbio's design, providing an overview on the most used feature extraction techniques for perception, discussing implementation issues. We intend to show the usefulness of a simulated platform as an inexpensive alternative for testing and developing computer vision algorithms in real-time robotics applications and its possible extensions to computer animated agents and avatars.

1 Introduction

A great deal of effort has been made recently towards developing new techniques (also improving existing ones) for the extraction of features from visual data (images) acquired by robotics vision platforms in restricted or open environments. The main issues are: what kind of features should be extracted from the environment and how to obtain them, how to define a feature, how to process the acquired images, how many time to look for a region in order to allow the system to get the desired features, and how to do all of this efficiently. All of these issues are still object of study in most of the computer vision and robotics research centers.

Usually, research in computer vision applied to robotics requires fragile and quite expensive pieces of machinery on which to perform experiments. Among this sort of equipment we may find cameras, small robots, and, mainly, stereo heads, which are robotic heads like the one shown in the top of Figure 1. Setting aside the cost of such dedicated architecture, the use of robots also restrict the development for the algorithms and experiments for the specific characteristics of that particular robot (number of DoFs, sensorimotor devices, etc). Also, it is not recommendable to perform any kind of experiment on a real robot; a dangerous task could compromise its (expensive) structure. For these main reasons, several algorithms are developed first in simulated devices, then placed into the real platforms.

In light of the above mentioned issues, this work presents a simulated device for robotics and humanoid research named *Câmbio*. With the use of such a simulated robot, research in Computer Vision and Robotics can be done at reduced

Figure 1: Current configuration of UMASS Torso "Magilla", gently offered by the Director of the Laboratory for Perceptual Robotics of UMASS. Thanks to David Wheeler, for risking his life:-). Picture used with permission.

costs, requiring only a computer and a freely avaliable graphical library. The simulated envirionment allow us to run different tests using different robotic setups. Also, we can test algorithms and situations that put the robot to its limits, without the fear of (or even intending) breaking it up. By using efficient simulation tools, we can work on various robotic fields, like cognition, attention, sensorimotor development, computer vision, and other fields, without the immediate need of a real robot. Mathematical models for those areas can be created and developed in these tools until they get mature enough to be tested "online". Of course, all of this come at some computer cost for modelling and running the simulated world. There are some associated costs not only related to running the world, but to the task of defining rules for such world, and to simulating it as well as the real world the robot will face later. We propose, however, that we could reach a reasonable amount of similarity to the "real world" if we put the simulated domain within certain bounds, and slowly stretch these bounds to upper limits.

In this work, we bound the simulated work to the domain of Computer Vision. We can thus provide a dynamic simulation of a stereo head and its image processing capability and of two integrated arms. Some of the tools for feature abstraction provided by the simulator Câmbio as the basis for high level processes (like, for instance, visual attention) are: use of Gaussian derivative operators for feature extraction, use of Multi Resolution Images (MR) for data reduction [9], use of Stereo Disparity and Motion features, and use of statistical moments. In the current version, we also offer two simulated arms to complement the visual information.

2 Related Work

The three dimensional simulator model proposed in this work, Câmbio, has carried out from the needs of developmental improvements to be done in another simulated robot environment, "Roger-The-Crab" [11, 10]. Roger is a bidimensional simulator imitating a crab which was originally proposed in 1991 to be used as a tool for low-level computer vision research. Since then, it has been used to develop tools and algorithms for many applications as for example attention control [7, 4], the study of meta-heuristics in machine learning algorithms for pattern categorization [6], the development of new models for computer animation [3], the development of tools for local perception in virtual animated agents [3]. Some of these works were the test-bed for real-time attention control and pattern categorization algorithms involved in robot cognition, which were later used in the online, real-world stereo head shown in Figure 1 [9, 8, 5]. Another important contribution was the development of computational models for robots, in the study of relationship between different sensorial systems like vision and touch to be used together on a close manipulation and grasping system [2]. The main improvement we propose on the previous version is the addition of the third dimension to the world model and also to the perception module. Other improvements as multi-processing, and other built-in capabilities as stereo disparity, vergence, gaussian and other convolution operators, are also been put on. We believe that this model will provide a much more accurate representation of the "real world" for systems that will later be implemented in online robots. This is a vital requirement, for instance, in the development of navigation algorithms and in the development of policies for humanoid robots based in visual information.

Visual attention can be defined as the ability to select a region of interest (a target) for extraction of information which could be useful to some task. A harder, related problem, is to add the ability of changing the attentional focus from one region of interest to another, be it due to changes in the environment or be it to acquire more, different data. In this work, the regions of interest are virtual objects determined by the mathematical model. This visual attention control is a measure of interaction between the robot and its environment. Some works[18, 12, 15] define only stationary images as the target of their attentional algorithms, not considering temporal aspects like motion, stereo, and behaviour. A common technique used in some works [14, 13] is the use of operators based as derivatives of the gaussian function for the extraction of features from a certain image (based on biological evidences [14]). A line of research based on explaining and imitating the working system of human biological eye is [17, 13]. In [13], they try to simulate a "fovea" region in which to put the zones of interest. To achieve this, multiple scales of a same image are generated, with the intent of centering the region of interest in the scale with bigger resolution. However, in general, these works do not usually apply to simulated robots, being instead developed in online robotic platforms. Another line of research related to our proposed simulator is the study of movements in virtual agents. Several works involving control of motion try to find techniques to generate realistic and precise behaviours for humanoid arms, legs, actuators. The use of inverse kinematics has been proved to be useful by many such works. In [1] it is used as a powerful tool in order to syntethise motion through the definition of special initial instances, and the interpolation of these instances to generate the final animation.

3 Theoretical background

The main developmental work done for this first stage of Câmbio was the design and implementation of the mainly used techniques for image processing in a simulated envi-

ronment. The main issue we had to take into account is that the computational platform will have to handle with the workload of executing a dynamic simulation, beyond performing the usual processing that takes place in an online robot. When taking such issues into account, we stress the importance of designing efficient implementations of the above mentioned algorithms, which can lead to new mathematical models for solving such problems in a better fashion.

3.1 Multi Resolution Retinal Images

In practice, to apply a complete low-level vision operator to an acquired image, it is generally necessary to perform several operations of "convolution". Each convolution to be applied to an n sided image through a p sided kernel (n and p being an integer amount of pixels) requires n^2p^2 multiplication operations. This yields a very bad result, specially when it is required to apply several operators in a sequence, in a given image, which is often the case in low-level vision. The use multi-resolution (often called multi-scale) images is a technique which tries to diminish this processing cost. This technique consists of using several smaller images (usually some 3 to 5) downsampled from the same original image. The images are obtained in such a way that they contain (in conjunction) the same (or little fewer) amount of visual energy (visual information) contained in the original image. The use of Gabor wavelets [16] has being one of the used approaches to generate such a multi-scale image representation in computer graphics and computer vision fields. In the current work, we use a simpler multi-resolution approach carrying out an image representation which is similar to the one present in the human eye, often called retinal image. In the human eye, the image captured by the retina is not sensed in an homogeneous manner. There is a central region in which occurs a higher concentration of structures for detection of color (light intensity). Therefore, the part of the image which lies in this region is perceived with greater detail than other parts of the image. This resolution falls off as we approach the periphery. Structures for detection of motion are more regularly distributed along the retina. This explains how we can see motion even for objects with image projected in regions far from the center of our retina. That is not true for object shapes (that need texture). Figure 2 illustrates a multi-resolution retina in which the most inner part (the right picture) appears in high resolution.

To transpose this idea to a computer-based domain, we'll take an image and reduce its size and resolution, getting a smaller representation of the same image. We achieve this through the use of an "average" filter on the original image. To produce the multi resolution image, we apply a number of these averaging filters with different kernel sizes

Figure 2: Multi-resolution image of a sphere. All images have the same number of pixels, in this case, $16 \times 16 = 256$ pixels.

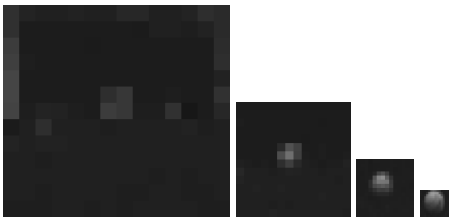

Figure 3: Multi-resolution image of a sphere. In the figure, they are re-scaled so one can note the region that each image covers on the original image.

through differently sized portions of the main image (so that each smaller portion is fully inside a bigger portion). This makes further algorithms and computations required by higher level processes easier. We'll end up with different images with different levels of resolution, representing different portions of our original image. The original represented portions can be seen in Figure 3, where the MR images in Figure 2 are re-scaled to show the region size that they cover on the original image. This directly relates to the "fovea" scheme presented above, for the final image with greater resolution shows a smaller portion of the original image, and each bigger portion is represented with lower resolution. Now, with these smaller images, we can apply other desired operators at a very reduced cost. Usually, for border detection and generic features extraction from the image (like light intensity and motion calculations), the lower resolution image will work fine for our needs. In some cases if we need higher resolution, as for instance for pattern matching, we can "foveate" the current region of interest, that is, put it inside the area of the smaller covered region, the higher resolution image. Then, process this higher resolution image which contains just a region of interest.

Gaussian operators

The gaussian operator is a convolution operator generated by a gaussian function applied to the size values of the bi-dimensional convolution mask to be generated. It is used in many algorithms for feature acquisition due to its very interesting property of "smoothing" the image. In general, this is a pre-processing phase before highlighting image borders. For different levels of smoothing and border de-

tection, the gaussian functions first and second derivatives are also often used as well. Ballard, in a recent work [13], has used up to the third order gaussian derivative as directional filters for feature detection, applied to recognition and attentional tasks. We have used up to the second order in previous work, for the same purpose [9]. In the current work, we provide up to the second order (Laplacian of Gaussian) Gaussian operator. Equation 1 represents the basic formula for the gaussian distribution. Equation 2 represents the gaussian convolution function and its derivative kernels are given by Equations 3, 4, and 5, in two orthogonal directions each. One can note we have modified the kernels of the derivatives, without loss of generality. This modification makes easier the implementation done, using integer values.

$$g_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-x^2/2\sigma^2} \tag{1}$$

$$G_{d=x,y}^{(k=0,1,2)} = g_d^{(k)} * I_t \tag{2}$$

$$\left.\begin{array}{l} g_x^{(0)}(x,y) = \lambda e^{ax^2} \\ g_y^{(0)}(x,y) = \lambda e^{ay^2} \end{array}\right\} \tag{3}$$

$$\left.\begin{array}{l} g_x^{(1)}(x,y) = 2a\lambda e^{a(x^2+y^2)}x \\ g_y^{(1)}(x,y) = 2a\lambda e^{a(x^2+y^2)}y \end{array}\right\} \tag{4}$$

$$\left.\begin{array}{l} g_x^{(2)}(x,y) = 2a\lambda e^{a(x^2+y^2)}(2ax^2+1) \\ g_y^{(2)}(x,y) = 2a\lambda e^{a(x^2+y^2)}(2ay^2+1) \end{array}\right\} \tag{5}$$

$\forall(x,y) \in ([-s,+s],[-s,+s])$; where $a = \frac{-1}{2\sigma^2}$, $\lambda = \frac{1}{\sigma\sqrt{2\pi}}$, $\sigma = 1.7$, and $s = 3$.

Stereo disparity

Stereo disparity can be defined as the perceived difference between the positions of an object (or a point) in two images generated by a pair of cameras (or eyes) set apart a certain distance from each other. There are several ways to calculate the stereo "disparity map", that is, a map where it is determined the disparity for each pixel in one image in relation to its corresponding in the other image. In this work, a simple correlation approach is used to determine disparity at each pair of images at each resolution level, over the second order derivative, considering estimations given by previous levels. By finding out the amount of stereo disparity between all points in the two images we can use the result to rebuild the three dimensional model that generated those images. We have showed in [9] that a multi-resolution stereo disparity map is enough for the purposes of attention control and object categorization, so we do not provide depth information (the reconstruction) in the current work. In order to accomplish attention control, we must choose a point in world space and try to put that point in both cameras, as to reduce stereo disparity to just what we need, then to extract three dimensional information. We provide an algorithm to do that through the "vergence" algorithm discussed above. This means centering the point in world space in one of the images (the "dominant eye"), and moving the other eye small amounts, while trying to minimize the stereo disparity around the central point in the dominant eye. The use of MR images can aid this task (which requires many calls to the "stereo disparity" function) by making it cheaper to estimate stereo disparity for a smaller image. The disparity of the lower resolution images is then used to estimate the disparity in the higher resolution ones, which are centered around the place where we wish to center our image. This scheme is shown in Figure 4.

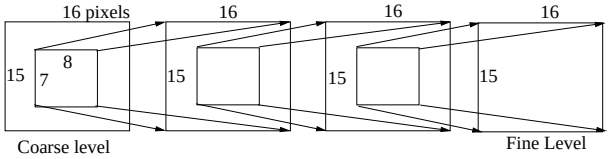

Figure 4: Stereo computation process in cascade.

Motion

Another use for the disparity map between two images is the retrieval of motion features in a dynamic scene. If the system wishes to turn its attentional focus into an area which contains moving objects, it can take a series of images obtained by a still camera, and figure out the disparity maps for them. Since the camera does not move, any disparity found will be due to mobile objects in the scene, being the disparity intensity proportional to the motion intensity. Here we compute motion separated for each image in the several resolution levels. That means, motion is computed as the disparity between the image taken in the instant t and the image taken in the instant $t-1$. In the same way as the above Gaussian images above, a MR motion image is produced. We prefer to use disparity as the measure of motion since this can solve most problems in attention control and other cognitive tasks. In the past works [9, 8], motion was computed as a simple image difference (what is enough for attention control).

4 Implementation

Figure 5 shows a screenshot of the interface. Câmbio was developed using a modular architecture. A multi-process application was generated so that each of Câmbio's modules can execute as a different process. The need to simulate a "real time world" leads to modular independence as an important requirement. Another nice feature we use is the ability to completely exchange one module for another

implementation of the same module, without touching any other parts in the simulator. Câmbio currently possesses four modules that seen in Figure 6: World Model Module, Sensory Servo Module, Decision Making Module and Control Module. The multi-processing used is a kernel-based one, for the GNU/linux operating system.

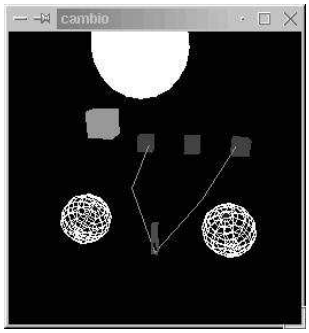

Figure 5: Câmbio's resources, including two eyes and two arms.

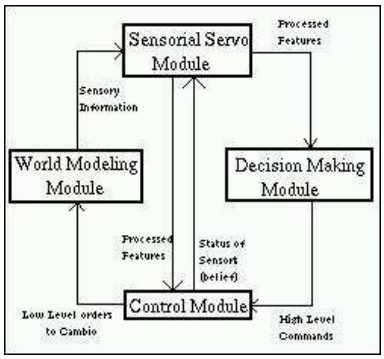

Figure 6: System modules.

World Model Module (WMM)

In Câmbio's architecture, the world is modeled in a simplified geometric fashion. Objects are stored through a set of their mathematical features. For instance, a sphere in a simulated room is stored by the values of its *key* index which identifies it uniquely among all objects in the world, its *type* index which defines what kind of primitive object (sphere, square, etc) it represents, the x, y integer world coordinates for the sphere's center, its radius' length value and color values. We are interested in further improving this world model with information of the object's physical properties. A special kind of object in the world models, in our current context, are Câmbios parts themselves. The world will store the parameters for these objects, and alter them according to the Control Module's requests. Currently, Câmbio possesses 3 kinds of "objects" in its design: neck with one degree of freedom (DoF) - rotating left and right around its axis; eye which is capable of acquiring sensory information and possessing 2 DoF and arm, possessing 4 DoF and a touch sensor at its end. Another role of the WMM is to produce sensory information as requested by the Sensory Servo Module. In this interface, the SSM asks the WMM to provide the readings for one of the robot's sensors. It provides the World Model with the parameters needed to check up the simulated world info and generate the sensorial answer. For instance, when the Sensory Servo Module wants to work on an image, it will ask for the sensory information of an eye, to the WMM, informing it from which eye it wants the information. The WMM will then use the eye's position and orientation to build the visual information and send it to the process running the sensory servo module. Currently, the acquisition of visual information is done by generating an OpenGL scene using the geometrical information of the objects of the world, and using the built in libraries to render the image based on the position and direction of the camera. The sensory information is acquired by a first round of "bubble" algorithm to find out which objects are candidate for "touching", and then geometrically testing each of these candidates against the end point of the arm. Since our initial concern is Visual feature extraction for attention, we do not worry about the arm "going through" objects now, but we will further provide this feature.

Sensory Servo Module

The Sensory Servo Module (SSM) must process the information received from the WMM, according to the requests of the Decision Taking Module (DTM). In the current stage of development, the SSM is a visual information server which processes images building feature maps needed for several, different algorithms to be used for high level processes as attention control. The sensory information for the arm, currently, is directly forwarded to the DTM without further processing. The SSM generates a Multi Resolution Image composed of 4 32×32 pixel images from the original 256×256 pixel image. The smaller images are generated through the MR algorithm described above, using 1, 2, 4 and 8 pixel-sided convolution masks for the averaging filter. It then applies any required convolution operators over this MR. Actually, we provide the partial derivatives of the Gaussian Operator (its zero, first, and second derivatives), for the σ values of 1.4, 2.7 and 4. The use of Gaussian operators has been proved to be useful for feature extraction as basis for attentional and categorization purposes [13, 9, 4]. We note that, in the current implementation, the use of the Gaussian function with other parameters, or even completely different convolution mask functions is possible with very little code changing. Also, it is offered the disparity operator which produces a multi-resolution disparity

map out of two images. This operator is used to compute the stereo disparity between the two current frames and also for detection of motion, between frames acquired in different time instants. Besides in the attentional behaviour, stereo will be also used in the vergence controller implemented in the Control Module, based on the disparity between the image acquired from both eyes at the same time. For motion detection, the server saves the image for a number of consecutive cycles and sends to the DMM the disparity between these images.

Decision Making Module

The Decision Making Module (DMM) is the most "user managed" of the modules. It uses the information acquired from the SSM, applies some sort of user defined algorithm in them, which results in actions to be forwarded to the Control Module. The DMM may contain learning, attention, and pattern recognition sub-systems based on the services provided by the other modules. Currently, we have implemented the upper level controller for the vergence algorithm, and the standard-input controls for moving each part of the robot and requesting information from the SSM. Other high-level tasks vary with what use is made of the simulator and will be inherent part of high-level algorithms further developed over the current architecture. Most of these high-level issues are not treated here, since they are not the purpose of the current paper.

Control Module

The role of the Control Module (CM) is to receive the high level actions from the DMM, and to turn them into lower level control directives for the robot actuators. For instance, while the DMM would say to the CM "move the right eye to the left", the CM would send that to the WMM as a "increase the angle $Vergence_1$ in the object "right eye" using servo controller C_{01} by 1 degree" command, which the WMM would in turn execute. The CM stores the same low level information about the "robot objects" as the world does, and alters them according to its actions. This redundancy exists to provide a certain degree of error to Câmbio, where the CM believes, by its actions, certain actuators to be in certain states, while in fact these states can be a little different due to real world errors. We currently implemented the existence of errors by using a simple gaussian noise operator applied to the controller which will effectively move Câmbio's physical resources towards a given target. Also, we have another error due to the discretization done to perform movements in the computer device. That is, a movement is broken into several differentiable but discrete sub-movements. So, at the end of a motion, some error will be present due to this discrete process (the resource will not be exactly on the target, but somewhere close to this. Note that by implementing this error simulation model, we approximate our device to a real platform, and that, differently of the real platform, we have ways to check where exactly a given resource is by using the information redundancy. This is useful in the development of high-level algorithms in commuter vision, since we have a high-controlled device. Finally, the Control Module takes charge of a few low level algorithms that can be ran "in background" for the DMM. These "background control tasks" are small, yet possibly complicated tasks that are useful for many different kind of algorithms.

Currently, we have implemented the "vergence controller", where the idea is to keep both eyes "verged" in one place in the world space while the robot searches for a new focus of interest, as in [4]. For the achieving and keeping of vergence on an point in space, we need to take two constraints into account. The first is the amount of degrees that each eye can move horizontally and vertically. We're developing simulator with a visual system based on the human physiology, thus we need to constrain the eye's movement. The eyes are attached to each other so that they cannot perform independent vertical movements. Also their horizontal movement has certain limits to a maximum degrees of vergence. Second, how to decide when Câmbio should move its eyes and when it should move its neck, and what are the motion parameters for each object in each direction, that is, what are the length and velocity/acceleration for each motion controller. With these constraints in mind, we propose to use a control architecture composed by two independent eye controllers, a coupled vergence controller and a main controller.

Figure 7 shows a scheme with the mechanical degrees of freedom of Câmbio's head. Each eye controller receives horizontal and vertical angle displacements from the vergence controller in order to position themselves in the environment. In the current setup, these angles are limited to 30 degrees in relation to each eye normal. If a movement requires for an eye to pass this limit, a neck movement has to be done in order for both eyes to reach the goal. The limit of the eye movement must be tinkered with until an optimal result can be reached.

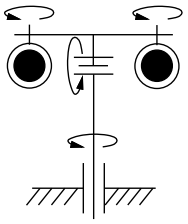

Figure 7: Mechanical DOFs in the Head.

The reason for allowing the eye controllers to having independent horizontal movement is that, usually, the fi-

nal vergence position requires different horizontal angles to the normal for each eye. The vergence controller aids the system by putting together the information obtained by each eye, and defining the central angle. The central controller is responsible for adjusting the neck position. The basic algorithm goes like the following. One of the eyes, given by some attentional process, is chosen to become the "dominant eye". This eye is then moved so that the vergence point is as close to the center of its image as possible. If this requires a horizontal movement greater than 30 degrees, the neck is moved along towards the target point. Otherwise just the eye moves. After the dominant eye is settled, the other eye's (called "non-dominant") position is adjusted in a loop which tries to minimize the disparity in the space around the vergence point. If this requires the non-dominant eye to perform a horizontal movement greater than its limits, the whole system is turned to allow more movement to the secondary eye (the dominant eye position is corrected). While doing all of that, the system also takes care for the eye axis to not pass the parallel to each other, when opening (besides this is physically possible even in humans we do not want for Câmbio to do that).

5 Experiments and demonstrations

In order to manually test the correlation algorithm for disparity working in the control of vergence, we put one of Câmbio's eyes still in an object point (in the right border of the sphere). Then we let the simulator to start its operating loop (the other eye is parallel to the first). The final eye images of this experiments are seen in Figure 8. In the top, besides a color, textured image is shown, we remember that every calculation is performed on the multi-resolution gray-level version of the object images (with texture) obtained by OpenGL. As a result, the other eye axis goes close to the first eye one, by minimizing disparity between the eye centers, that means by maximizing correlation between the images. This is a very good visual result for this textured image. One can note depth from the images by verging the eyes, one in each picture. The same experiment with a cube (a not textured object, only using shading for the correlation) is seen in the bottom of Figure 8. In this case, the algorithm could verge close to the object, but it did not perform that well due to the lack of texture. To minimize this, one can include more objects in the scene.

Next we move Câmbio back in its environment, far from the object, keeping the same point in space for the right eye (the cube). It keeps verged in the same object and other objects are put in the scene, entering its visual field, as seen in Figure 9. The correlation approach and the sparse distribution of the objects in the environment makes these objects interfere a little in the vergence. Besides, depth can also be noted here by verging the eyes.

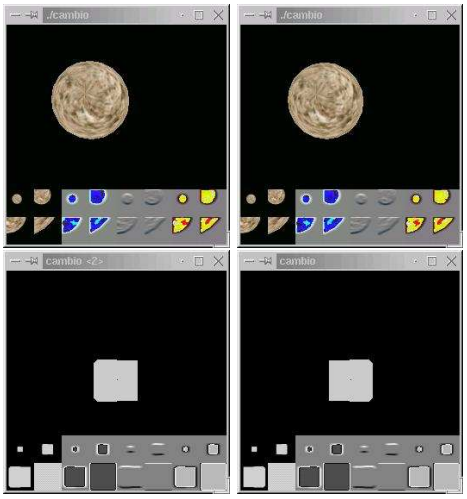

Figure 8: Letting Câmbio's eye verge on 3D objects.

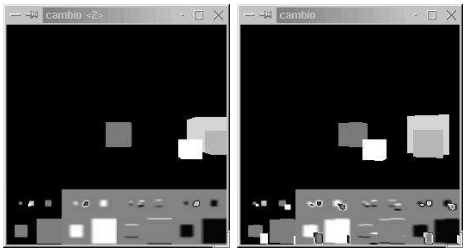

Figure 9: Câmbio gets back and verge again.

In other tests for the vergence algorithm, we put Câmbio in execution with its eyes initially in parallel, then wait for it to verge and send some eye motion commands to its control system (this is called a saccadic movement in the same way as our biological eyes do). That is, Câmbio had to move from point to point in the scene. Figure 10 shows the initial situation for both eyes (verged in a cube). An angular displacement is calculated and passed to its control software in order to move to other region (to the cube on the left part of Figure 10). We could see that, while moving, its eyes can not stay verged due to the motion algorithm used (one eye following the other). But, when the dominant eye stops the movement (reaches the target), the other eye verges immediately by correlating the images, now in the other cube. Figure 10 shows the initial state of the eyes (looking to a certain place) and the final state of this sacaddic motion. One can note stereoscopy on the environment by fixing one eye on each of the images. We note that there is no attentional algorithm operating here, so Câmbio is not exactly verged in any object, but in a given target entered by the operator. Several attentional policies will be implemented later, as other research work, based on the current features defined.

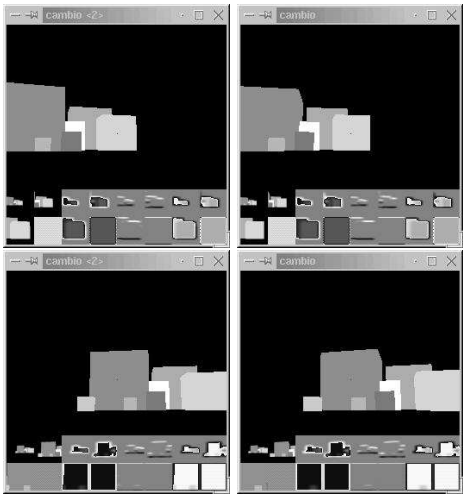

Figure 10: Câmbio executing a saccadic movement.

6 Conclusions and future work

Currently, we have the Gaussians, averaging, disparity and Sobel's operators implemented in Câmbio by using OpenGL ("C") directives. It is our aim to add as many different operators as possible, so that one can try different approaches to feature extraction and higher level processes. We intend to perform comparative studies of such operators using Câmbio's architecture. We also plan to add more complex objects to the world model and implement some more realistics physics laws to the world simulation, like the ability to bump into objects and, consequently, develop a policy to perform motion in such a populated environment. Finally we intend to implement, test and compare different attentional and feature extraction learning systems, like those presented in the references, as this is the main goal used for the development of this simulation.

References

[1] J.-S. M. P. Baerlocher, R. Boulic, and D. Thalmann. Using an intermediate skeleton and inverse kinematics for motion retarding. *The Eurographics Association and Blackwell Publishers 2000*, 2000.

[2] S. Botelho and L. Garcia. Combining local perception, low-level manipulation and task achievement control in multi-robot teams. In *Proceedings of International Joint Conference IBERAMIA/SBIA'2000*, pages 246–255, Atibaia, SP, Brasil, November, 19-22 2000. Springer Verlag.

[3] L. Gonçalves and F. Silva. Control mechanisms and local perception to support automomous behavior in virtual animated agents. *Computer & Graphics Journal*, 25(6):965–982, December 2001.

[4] L. M. G. Gonçalves. Towards a learning model for feature integration in attention control. In *Proc. of IEEE MultiSensor Fusion Conference (MFI 2001)*, Bade-Baden, Germany, Ausgust, 19-22 2001. IEEE Press.

[5] L. M. G. Gonçalves, C. Distante, A. Oliveira, D. Wheeler, and R. A. Grupen. Neural mechanisms for learning of attention control and pattern categorization as basis for robot cognition. In *In Proc. of International Intelligent Robots and Systems Conference (IROS 2000)*. IEEE Computer Society, October 30 - November 5 2000.

[6] L. M. G. Gonçalves, R. C. Farias, F. W. Silva, S. C. Botelho, and A. A. Lopes Junior. Control mechanics for computer animated agents. In R. S. W. Luciana P. Nedel, editor, *Proc. of III Brazilian Workshop on Virtual Reality*, pages 193–204, Porto Alegre, RS, October, 16–18 2000. Gráfi ca EPECÊ / Pontifícia Universidade Católica do Rio Grande do Sul.

[7] L. M. G. Gonçalves, G. A. Giraldi, A. A. F. Oliveira, and R. A. Grupen. Learning policies for attentional control. *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA '99)*, November 1999.

[8] L. M. G. Gonçalves and R. A. Grupen. Integrating attention and categorization behaviors in robotics. In *Proc. of the 6th International Conference on Intelligent and Autonomous Systems*. IEEE Computer Society Press, July, 25-27th 2000.

[9] L. M. G. Gonçalves, R. A. Grupen, A. A. Oliveira, D. Wheeler, and A. Fagg. Tracing patterns and attention: Humanoid robot cognition. *The Intelligent Systems and their Applications*, 15(4):70–77, July/August 2000.

[10] L. M. G. Gonçalves and F. W. Silva. Task manipulation and control in artifi cial animated creatures. In M. Berthoz Floreano Roitblat Wilson, editor, *Proc. of "From Animals to Animats" - VI International Conference on the Simulation of Adaptive Behavior (SAB'00)*, pages 77–86. ISAB Inc., September, 11-15th 2000. Conference held in Paris, France.

[11] L. M. G. Gonçalves, F. W. d. Silva, R. C. Farias, and R. A. Grupen. Towards an architecture for artifi cial animated creatures. In *Proc. of VI CGS/IEEE Computer Animation Conference*, pages 170–175, Los Alamitos, CA, May, 3-5 2000. IEEE Press. Conference held at Philadelphia, PA.

[12] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Patten Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.

[13] M. M. H. Rajesh P.N. Rao, Gregory J. Zelinsky and D. H. Ballard. Eye movements in iconic visual search. *Vision Research*, 2002.

[14] R. P. N. Rao and D. Ballard. An active vision architecture based on iconic representations. *Artificial Intelligence Magazine*, 78(1-2):461–505, October 1995.

[15] I. A. Rybak, V. I. Gusakova, A. V. Golovan, L. N. Podladchikova, and N. A. Shevtsova. A model of attention-guided visual perception and recognition. *Vision Research*, 38(2):387–400, 1998.

[16] T. D. SANGER. Stereo disparity computation using gabor fi lters. *Biology and Cybernetics*, 59:405–418, 1988.

[17] J. Tsotsos, S. Culhane, W. Wai, Y. Lai, N. Davis, and F. Nuflo. Modeling visual attention via selective tuning. *Artificial Intelligence Magazine*, 78(1-2):507–547, October 1995.

[18] P. Van de Laar, T. Heskes, and S. Gielen. Task-dependent learning of attention. *Neural Networks*, 10(6):981–992, August 1997.