A Humanoid Robot Drawing Human Portraits*

Sylvain Calinon, Julien Epiney and Aude Billard
Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland
{*sylvain.calinon, julien.epiney, aude.billard*}*@epfl.ch*

***Abstract*— This paper presents the creation of a robot capable of drawing artistic portraits. The application is purely entertaining and based on existing tools for face detection and image reconstruction, as well as classical tools for trajectory planning of a 4 DOFs robot arm. The innovation of the application lies in the care we took to make the whole process as human-like as possible. The robot's motions and its drawings follow a style characteristic to humans. The portraits conserve the esthetics features of the original images. The whole process is interactive, using speech recognition and speech synthesis to conduct the scenario.**

***Index Terms*— Face detection, image processing, humanoid robot, human-robot interaction, entertaining robot, trajectory planning, inverse kinematics**

I. Introduction

The development of humanoids is motivated by our inner wish to build human-like creatures. Humanoids should resemble us both physically and "intellectually", displaying human-like behaviors. Humans do not always behave purposefully. A non-neglecting part of our time is devoted to entertain ourselves. Thus, while we may provide humanoids with the capacity to perform useful tasks, we may also want those to entertain us. Across the centuries, humans have shown an eager interest in having their portraits drawn, whether for satisfying their own ego or for leaving a trace to the posterity. There are many ways to draw one's portrait and, while caricatures may at times be popular, embellishing portraits that keep the esthetics, leaving out the "unnecessary" features, remain most appreciated. Various techniques for image processing have slowly crawled their way into our everyday making of portraits (think only of the numerous digitized images blasted by the media).

In this paper, we present an application of these techniques to provide a humanoid robot with the ability to draw esthetic portraits. This follows a recent trend that aims at developing entertaining behaviors in humanoid robots [1]–[3]. The behavior of the robot resembles that of an artist, who would, first, analyze the face of the model, and then, draw the portrait on a piece of paper. The robot may, also, ask for a pen and take it from the user. The application runs in real-time and is interactive. It uses existing tools for image processing, robot control, object manipulation and speech processing.

Sketching a face requires human-specific skills, and is thus a challenge for humanoid robotics. Prior attempts at creating a painting robot can be found in the robotics literature, see, e.g. [4]–[6]. However, none of these works used a humanoid robot, nor tried to make the painting artistic and the drawing process look human-like. These authors used either an industrial robotic arm or an XY-type planar robot. Moreover, the drawing process resembled that of a printer, i.e. reproducing horizontal lines from top to bottom.

Visual analysis of human faces draws from a large body of work in image processing. To improve the efficiency of face recognition algorithms, a transformation of the data space is usually required. This transformation leads to a representation of the image which is not meaningful to a human user, e.g. spectral analysis. In the present work, we extract the main characteristics of a face, making sure that the resulting representation may still be recognizable as a human face. Thus, the criteria for our face analysis are not based on recognition efficiency, as it is usually the case, but on the "esthetic", required for our particular application.

A number of prior works have followed a similar approach to extracting artistic features of a face, by matching those with a library of stylized cartoon-like features, see, e.g. [7]. The goal was, then, to create a digital avatar from a picture facing the camera, and required the intervention of the user at different steps during the process through a computer-based interface. We follow a similar approach, but, depart from a PC-based interface to provide natural human-machine interactions, letting the user completely unaware of the underlying processing, and making sure that the system is robust to changes in lighting condition (shadows) and orientation of the head with respect to the camera.

The rest of the paper is organized as follows: Section II presents the system set-up, the tools used, and the scenario. Section III presents the processes of image analysis, path planning and robot control. Section IV discusses the results and shows the possible extensions of the system.

II. Experimental set-up

A. Hardware and software

The experiments are performed on a *Fujitsu* humanoid robot HOAP2 with 25 degrees of freedom (DOFs). Note that only the robot's right arm (4 DOFs), right hand (1 DOF) and head (2 DOFs) are used in this experiment. The arm consists of 3 rotations for the shoulder (flexion-extension, abduction-adduction and humeral rotation), and 1 for the elbow. The hand gripper has 5 fingers, actuated by a single motor. The torso and legs are set to a constant and stable position, in order to support the robot sitting with its torso in a upright position. An external Phillips webcam tracks the head of the user at a rate of 15Hz and with a resolution of 640x480 pixels. A sound

*This work is partially conducted within the EU Integrated Project COGNIRON and is supported in part by the Swiss National Science Foundation, through grant 620-066127 of the SNF Professorships program.

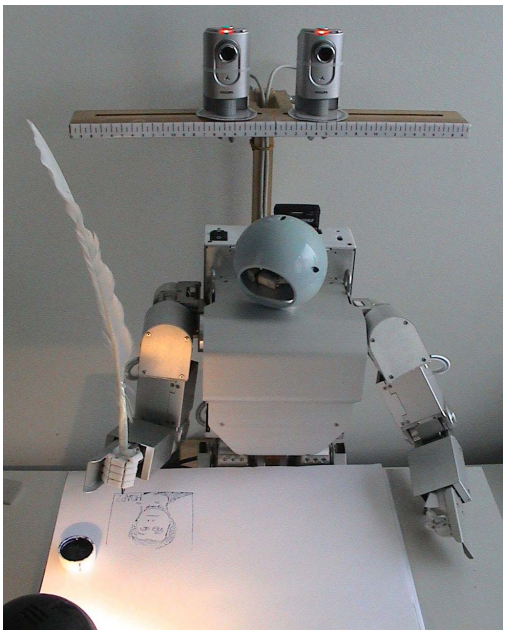

Fig. 1. Experimental setup.

speaker system, developed in our lab, is located in the robot's head, to provide synthesized speech. A standard microphone is placed close to the robot.

The Fujitsu HOAP2 controller library is used to control the robot. *OpenCV* is used to grab data from the webcam and to process the images [8]. *Festival* is used for speech synthesis [9]. *Sphinx-2* is used for speech recognition [10]. All of these softwares have access to their source-codes and run under a Linux environment.

B. Experimental scenario

A table is placed in front of the robot, see Fig. 1. A vision system detects if a user is facing the robot. When the user is approaching, the robot turns its head toward the user, and asks him/her if he/she wants his portrait drawn by a robot, using its speech synthesis system. If a positive answer is detected by the speech recognition system, the robot takes a snapshot of the user's face, and asks for its quill pen. The robot moves its right arm in the direction of the user. The user asks the robot to open its hand. He/she puts the quill pen in its hand, and asks the robot to close its hand. The robot, then, grasps the pen, turns its head back to the table and starts drawing the portrait of the user on a piece of paper. Instead of drawing the portrait in the manner of a printer, the robot sketches the face naturally, as a human would do. It first starts by drawing the contours, starting with the rough silhouette, and adding details iteratively. Then, it fills the dark areas using different filling styles. From time to time, depending on the length of the trajectories used to draw the portrait (i.e. the level of ink in the quill pen), the robot uses an inkpot to refill the pen. Once the drawing has been completed, the robot signs the drawing. The scenario is represented in Fig. 2.

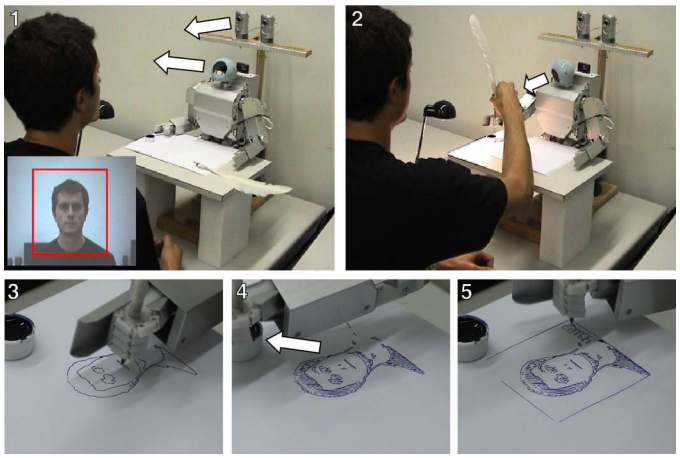

Fig. 2. Experimental scenario. *1)* The vision system detects the presence of a user (see inset), takes a snapshot of his face, while the robot turns its head toward the user. *2)* The robot moves its arm toward the user and asks for its quill pen, using speech synthesis. Speech recognition is used to give simple instructions to open and close the robot's hand, so that it can grasp the pen hold by the user. The robot grabs the pen, looking at its hand. *3)* The robot starts sketching the portrait of the user, starting with the rough contours, and adding details iteratively. *4)* It then fills the areas bounded by the contours. When the quill pen is empty, i.e. when the length of the trajectory has reached a given threshold, the robot uses an ink-pot to refill the pen. *5)* Finally, the robot draws a frame around the portrait, signs its drawing, turns its head to the user, and signals verbally that the portrait is finished.

III. Data processing

A. Face tracking

Haar-like features tracking is used to detect faces in the image, as in [11]. It is based on features tracking using a gradient information characterizing the different parts of a face (e.g. nose, eyes). As the tracker detects only the face, it creates an image subspace $\mathcal{R}_1$ surrounding the features. In order to include the hair and the shoulders of the user in the portrait, another subspace $\mathcal{R}_2$ is defined, centered on $\mathcal{R}_1$, whose size is 3 times larger than $\mathcal{R}_1$ in both directions. If multiple faces are detected, the closest person will be selected, i.e. the one with the largest surrounding window. $\mathcal{R}_1$ and $\mathcal{R}_2$ are then rescaled to a fixed size.

B. Contour extraction

A *Canny* edge detector is used to detect the positions in the image showing intensity discontinuities [12]. The basic idea is to detect the zero-crossings of the second directional derivative of the smoothed image. It produces in the output a thin line that can be used easily to compute a trajectory (see Fig. 3). The closed contours are coded as a *Freeman* chain code, starting from a random position. They are ordered according to their lengths.

C. Image binarization

For each pixel in the image $\mathcal{R}_2$, the luminosity value $Y \in \{0, 1, \ldots, 256\}$ (from black to white), from the YUV encoding scheme is transformed to a binary value $\{0, 1\}$ (black and white), following a thresholding method. It converts the image to a *black & white* representation that can be drawn by the

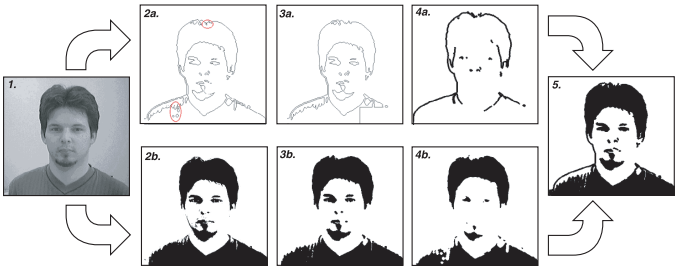

Fig. 3. Image processing. *1)* Image $\mathcal{R}_2$ extracted by the *Haar* face detector. *2a)* Contours extracted by the *Canny* edge detector. *2b)* Binarized image extracted by applying a threshold keeping 23% of black pixels in image $\mathcal{R}_1$. *3a)* Contours with irrelevant features discarded using Equation 2. A white area is added at the bottom-right of the image, to let a free space for the signature. *3b)* Different morphological filters are applied to the image to discard irrelevant noise. *4a)* An image is created from the contours image *3a* by applying morphological filters, to make the trajectory appear wider. *4b)* The image *4a* is subtracted from the image *3b*. The resulting image corresponds to the areas that will be filled with ink. *5)* Preview of the final result, by superimposing images *4a* and *4b*. The robot first draws the contours as in image *4a*, and, then, fill the areas as in image *4b*.

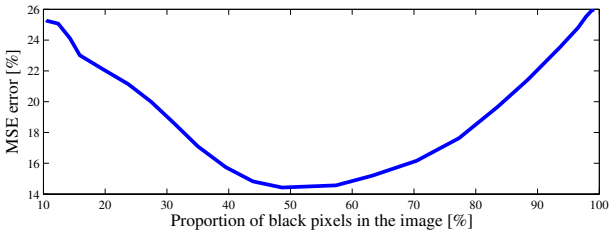

Fig. 4. Reconstruction error in function of the proportion of black pixels in image $\mathcal{R}_1$. The minimum error is found approximately when half of the pixels are black.

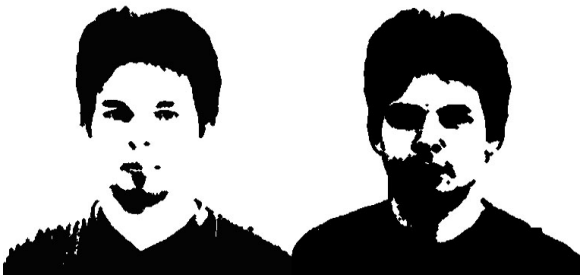

Fig. 5. Image binarization using different thresholds. *Left:* Using an "esthetic" criterion defined by a human user on the basis of a few examples ($\alpha = 23\%$). *Right:* With a minimum reconstruction error criterion ($\alpha = 50\%$). Although the reconstruction error criterion is mathematically the transformation losing least information, we see that it is not optimal from an esthetic point of view, i.e. the features in the left inset are separated more properly.

Fig. 6. Test of the binarization process with images files. The criterion is based on a fixed proportion of black pixels in the area surrounding the eyes and the lips. We see that the criterion is quite robust to non-homogenous lighting conditions (*left*), dark skin color (*middle*), and light hair color (*right*).

robot using black ink on white paper. To select the threshold, a criterion must be defined. The criterion, used most frequently, is to minimize the mean-square error (MSE) between the two representations. In Fig. 4, this reconstruction error has been computed on the basis of 3 different images, with respect to the proportion of black pixels retained during the thresholding process. As expected, the minimum error is found around 50%, which means that keeping half of the pixels black is the best transformation, with respect to a MSE criterion. However, by looking at Fig. 5, right, we see that it is not the most satisfying condition. It produces binarized images that have too many black pixels connected to one-another to recognize a face correctly. Indeed, the essential features are merged together and are not distinguishable anymore, i.e. the eyes, eyebrows, lips, nose and neck are connected and represented by a single area of black pixels. To address this issue, we determined a factor α, that fixes the proportion of black pixels in the subspace $\mathcal{R}_1$ after transformation. It ensures that in the area surrounding the essential features of the face, the proportion α of black pixels is fixed, making sure that the eyes, lips, nose and eyebrows are distinguishable in the majority of the cases. The proportion of black pixels has been fixed experimentally by a human observer, on the basis of 15 faces examples. To fix the threshold, a grayscale histogram $\{n_0, n_1, \ldots, n_{256}\}$ is created from the image $\mathcal{R}_1$, where n_i denotes the number of pixels of luminosity value i, in the image $\mathcal{R}_1$. A color threshold T is selected by finding the highest value of T satisfying:

$$\frac{\sum_{i=0}^{T} n_i}{\sum_{i=0}^{256} n_i} < \alpha \qquad (1)$$

After transformation, the image $\mathcal{R}_1$ will then roughly consist of a proportion of α black pixels. $\alpha = 23\%$ has been fixed experimentally (see Fig. 5, left). The grayscale threshold T is then used to binarize the whole portrait image $\mathcal{R}_2$. This method has the advantage to be robust to changes in luminosity, and to focus on the essential features of the face to select the threshold. Thus, the thresholding criterion does not depend on the background, hair color, or clothes of the user. The robustness of the method to process different faces is illustrated in Fig. 6. We see that the adopted criterion is quite robust to non-homogenous lighting conditions, dark skin color, and light hair color.

Salt & pepper noise was, then, removed by applying morphological filters to the binarized image [13]. One *close* operation (*erosion-dilation*) is performed, followed by two consecutive *open* operations (*dilation-erosion*), using a 5 pixels mask. The two operations are aimed at removing isolated black pixels in white areas, and vise-versa. The two consecutive *open* operations ensure that the level of details of the resulting image corresponds roughly to the resolution of the pen used by the robot, i.e. the thin elements can still be drawn.

D. Preparing the data to draw

As mentioned in introduction, the drawing process will be divided into two phases. The robot will start drawing the

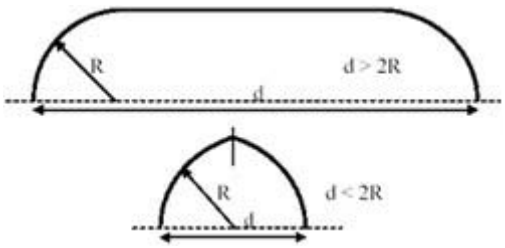

Fig. 7. *Top:* Trajectory transition, parameterizable to go from one point to another, without touching the paper with the pen. *Bottom:* If two points are too close, the trajectory is truncated.

contours, and, then, will fill all the dark areas. In order to achieve this, we divide the resulting image into two datasets, containing the contours and the parts to be filled in. To do so, we, first, removed the contours, found by the method described in Section III-B, from the binarized image to obtain distinct non-connected areas, corresponding to the areas to fill with ink, see Fig. 3.

An usual technique to get rid of the irrelevant contours in an image is to discard the contours whose surface is below a fixed threshold. We would like to get rid of the irrelevant features in the background, while keeping the small features characterizing a face. Learning the background usually requires fixed camera and still background, which does not fit in our application, where several persons can stand in front of the robot. We propose an alternative to the simple thresholding approach, by taking into account the position of the center of gravity of each contour, relative to the center of the image, i.e. the nose. A linear relation is sufficient in our case, to weight the surface with its position on the image, i.e. to satisfy the criterion:

$$w(M_{ij}) \cdot M_{00} > T_S \tag{2}$$

with

$$w(M_{ij}) = \frac{1}{\sqrt{(\frac{M_{10}}{M_{00}} - \frac{W}{2})^2 + (\frac{M_{01}}{M_{00}} - \frac{H}{2})^2}} \tag{3}$$

where T_S is the threshold, M_{ij} is the 2-dimensional geometric moment of order (i+j) [14], i.e. M_{00} is the surface of the contour, while $\frac{M_{10}}{M_{00}}$ and $\frac{M_{01}}{M_{00}}$ are the centers of gravity along the two dimensions. W and H are the width and height of image $\mathcal{R}_2$. In our experiment, $T_S = 0.7$ and $W = H = 400$ have been fixed. This equation ensures that most of the features characterizing the face (i.e. close to the nose) will be kept, while the other features in the image will be progressively withdrawn, if their surface is too small.

E. Trajectory planning

The first step is to define the working area where the robot will draw the portrait. In simulation, using the characteristics of the robot, and an inverse-kinematics model of its right arm (see Section III-G), an optimal planar workspace is searched, in order to maximize the drawing area, given the robot architecture. The search is an off-line process, where an horizontal rectangular trajectory is iteratively displaced in the three dimensions, and scaled in the 2 dimensions of a horizontal plane. At each iteration, the inverse-kinematics model checks if the trajectory can be achieved by the robot. The area defining the maximum surface is selected as the drawing plane, which corresponds to a natural writing position (see Fig. 1).

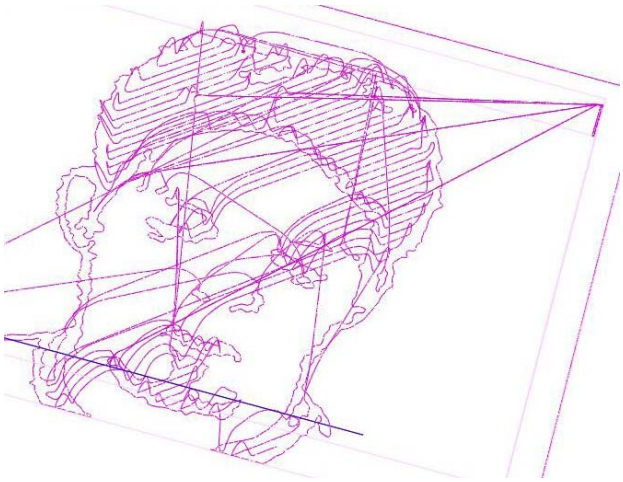

Fig. 8. 3D trajectory of the right hand in Cartesian space. The transitions between the strokes are depicted, as well as the trajectories used to refill the pen (the ink-pot position is on the top-right corner of the drawing).

The contours are first drawn by the robot, starting from a rough silhouette, and adding iteratively smaller details. To do so, it draws the contours according to their *Freeman* code length, starting from the longest one. The *Freeman* code is transformed to a 2D trajectory, and is interpolated using a 3rd order spline fit to smooth the trajectory. It avoids the *pixels-effects* on the drawing, and the small jerks and vibrations when the robot's hand is moving from one pixel to another, appearing when the trajectory is not smoothed. It also allows to add a desired number of points to the trajectory, since the controller of our robot requires one command every millisecond, i.e. the speed is directly related to the number of points. The number of points in the interpolation is calculated such as the final drawing trajectory played by the robot will last no more than 2 minutes. The 2D trajectory is automatically rotated and rescaled to fit in the best drawing plane found previously.

To transit from one contour to another, the robot lifts the pen, goes straight to the next point, and puts the pen back on the piece of paper. A 3rd dimension is added to the 2D trajectory to make the transitions in a vertical plane. To look more natural, the transition trajectory is smoothed, as shown in Fig. 7. Depending on the distance between two points, the trajectory is truncated, i.e. the robot lifts the pen adequately if two points are too close. The 3D Cartesian trajectory is shown in Fig. 8.

F. Drawing modes

Each area is, then, filled with a drawing pattern, from the largest to the smallest area. 3 different modes can be selected, and defines the drawing style of the robot: hatching in one

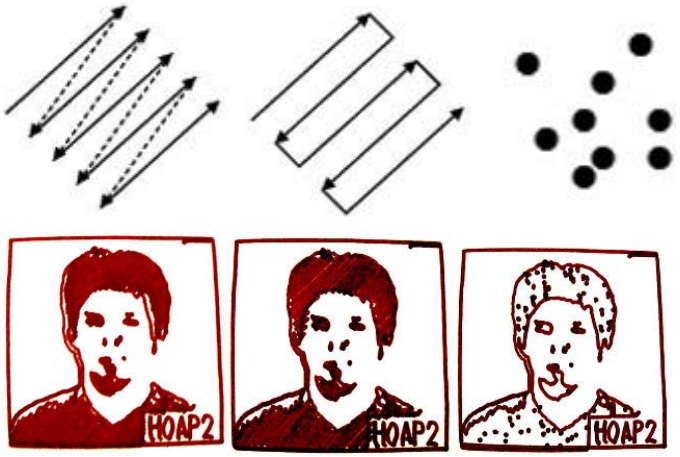

Fig. 9. *Top:* Different parameterizable trajectories used to fill the drawing with different styles of filling patterns. *Bottom:* Resulting drawing. The hatching in one direction gives the best results, but is 30% longer than the hatching in two directions, due to the transitions. The last mode uses a pre-defined pattern (here, a single dot), and fill the areas with a desired density of patterns, whose positions are selected randomly.

Fig. 10. *Top:* Faces detected from the webcam (subspaces $\mathcal{R}_2$). *Bottom:* Final portraits drawn by the robot, using different kinds of pen (marker pens and quill pen).

direction, hatching in two directions, and filling with patterns at random positions. These modes are depicted in Fig. 9. The first two modes are tracing lines to fill the areas, whose distances and orientation are parameterizable. The hatching in one direction results in a more precise and homogenous filling than the hatching mode in two directions (some white lines remain in the filling areas, see middle inset in Fig. 9), but the resulting trajectory is also about 30% longer, due to the transitions. The last mode selects a number of random positions in the area, and draws patterns at these positions. The number of patterns is parameterizable, and proportional to the size of the area. The pattern to use is also parameterizable, defined as a trajectory (the default pattern is a single dot).

The lengths of the trajectories, when the pen is in contact with the paper, are summed to estimate the level of ink used. If it is up to a threshold, the robot stops drawing, moves its hand to an ink-pot fixed on the table, with a transition trajectory adapted from the one in Fig. 7, and comes back to where it has stopped.

At the end, the robot draws a frame around the portrait, and signs its drawing. To do so, a collection of predefined trajectories corresponding to letters is used. Given the name of the robot (here *HOAP2*), the signing trajectory is computed automatically, and drawn at the bottom-right corner of the frame[1], see Fig. 10.

G. Robot controller

We compute the joint angle trajectories of the drawing arm (right arm) of the robot, based on the Cartesian trajectory of the right hand holding the pen. We must take into account the body constraints of the robot, which are given by the inverse kinematics function $\theta = f^{-1}(x)$. We consider an iterative, locally linear, solution to this equation, such that:

$$\dot{x} = \mathbf{J} \cdot \dot{\theta} \tag{4}$$

where $\dot{\theta}(t) = \theta(t) - \theta(t-1)$ and $\dot{x}(t) = x(t) - x(t-1)$ are the velocity vectors of the joints and the hand path. $\mathbf{J}$ is the Jacobian, a 4×3 matrix.

A solution to this equation can be found using the *pseudo-inverse with optimization* numerical solution [15]:

$$\begin{aligned} \dot{\theta} &= \mathbf{J}^{+}\dot{x} + \alpha(\mathbf{I} - \mathbf{J}^{+}\mathbf{J})g(\theta) \\ \mathbf{J}^{+} &= \mathbf{J}^{T}(\mathbf{J}\mathbf{J}^{T})^{-1} \\ g(\theta) &= \theta_r - \theta \end{aligned} \tag{5}$$

where $g(\theta)$ is an optimization term which tends to minimize the distance between the arm posture θ and a rest position θ_r. θ_r is defined off-line as a natural position of the arm, when the robot sits in front of the table with its forearm lying on the table. The contribution of this constraint, defined by $\alpha = 0.9$, is fixed experimentally.

Inverse kinematics is also used to control the head of the robot, so that the robot appears to look at the piece of paper, at the ink-pot, or at the head of the user. The PID controller of the robot, provided by Fujitsu, is then used to send motor commands to the robot.

IV. Results, discussion and further work

Some results are presented in Fig. 9 and 10. Drawing with a quill pen is not an easy task, even for humans. When first experienced with quill pen drawing, novices usually leave stains on the paper, due to unappropriate gestures when manipulating the pen in the air, during the transitions. Here, the robot performs the task skillfully. By planning a smooth trajectory, controlling the speed and acceleration of the arm, the robot hold the pen filled with ink, without losing ink at unappropriate places.

We see that the lines to draw the frames around the portraits are not really straight and parallel. It is mainly due to errors

[1] A white area was previously superimposed to image $\mathcal{R}_2$, to ensure that the sign will remain visible.

in the model of the robot used during the inverse-kinematics process. More specifically, the position of the pen is supposed to be fixed, but it can change from one example to another, depending on the position of the pen given to the robot. This position can also slightly change during the drawing. Errors are also introduced in the set-up (e.g. if the robot does not sit properly, or if the table is not perfectly horizontal).

The HOAP2 robot is not actuated at the level of the wrist. It is a severe drawback for object manipulation and writing. When the robot's hand is close to the limits of its writing area (e.g. when drawing the frame), the pen is not perpendicular anymore, due to the missing degree of freedom.

Each step of the whole process can be monitored and controlled visually, and parameters can be changed online, if desired. However, the application is created to run without the intervention of the user. The whole process is fast: the image processing part and inverse kinematics take only a few seconds, and the drawing takes less than 2 minutes. A similar portrait sketched by a human would clearly take more time to draw. A video of the drawing process is available at `http://humanoids.epfl.ch/`.

In further work, we plan to add more interactive behaviors to the robot. E.g. the robot may ask the user whether he/she would like to have some more personalized addition been made to the drawing. The stereoscopic vision system developed in our lab could, also, be used to make the face tracking more robust, and to eliminate the details in the background.

The process of transforming the image in a *black & white* representation could be improved in many ways. For instance, using a thresholding process, other criteria such as a *solidity* measure could be considered, to better distinguish the facial features. *Solidity* is a measure of a surface divided by the convex surface surrounding the group of pixels. It could, then, be possible to detect if multiple features are merged together, resulting in non-convex areas.

Instead of transforming the image in two colors, more shades of gray could be considered. Depending on the shade of gray, the robot could fill the areas with different patterns of different densities, e.g. by adjusting the density of patterns in the third mode, or by varying the space and orientation of the lines in the first two modes (see Fig. 9). Another solution, that will be considered in further work, is to use ink-pots with different types of ink, or, similarly, by allowing the robot to use several pens of different colors.

Considering only a *black & white* image, more sophisticated computer graphics techniques could also be considered. Indeed, half-toning, dithering, stippling, digital engraving or other non-photorealistic rendering techniques can be considered as more powerful extensions to the binarization problem, see e.g. [16]–[18].

V. Conclusion

This paper presented a simple realization of a robot capable of painting realistic portraits. While the methods we used are not novel, their integration in this demonstration is unique. The contribution of this work to humanoid robotics is two-fold: first, it demonstrates how image processing techniques and classical trajectory planning algorithms can be tuned to endow a humanoid robot with human-like competencies; second, it highlights the potential that small humanoid robot may have as an entertainment device. Such a small robot drawing dutifully on a table bears much resemblance to a child drawing its first picture: an analogy that may be much harder to elicit when using an industrial robotic arm.

Acknowledgment

We would like to thank Xavier Perrin, MSc student, for his contribution to the development of the speech processing interface, and Florent Guenter, PhD student, for his contribution to the development of the inverse kinematics module.

References

[1] M. Fujita and H. Kitano, "Development of an autonomous quadruped robot for robot entertainment," *Autonomous Robots, 5*, pp. 7–18, 1998.

[2] M. Fujita, Y. Kuroki, T. Ishida, and T. Doi, "A small humanoid robot sdr-4x for entertainment applications," in *IEEE/ASME Intl Conference on Advenced Intelligent Mechatronics*, 2003.

[3] A. Billard, "Robota: Clever toy and educational tool," *Robotics & Autonomous Systems*, vol. 42, pp. 259–269, 2003.

[4] M. Stein and C. Madden, "The pumapaint project: Long term usage trends and the move to three dimensions," in *Proceedings of the IEEE Intl Conference on Robotics and Automation (ICRA)*, Barcelona, Spain, April 18-22 2005.

[5] M. Gommel, M. Haitz, and J. Zappe, "Robotlab autoportrait project: Human portraits drawn by a robot," 2004, center for Art and Media Karlsruhe. Available at http://www.robotlab.de.

[6] J. Lehni, "Hektor: a graffiti output device," 2002, diploma project at the Ecole Cantonale d'Art de Lausanne. Available at http://www.hektor.ch.

[7] H. Chen, Z. Liu, C. Rose, Y. Xu, H.-Y. Shum, and D. Salesin, "Example-based composite sketching of human portraits," in *Proceedings of the international symposium on Non-photorealistic animation and rendering*, 2004, pp. 95–153.

[8] Intel Research, "The opencv open source computer vision library," 2005, available at http://www.intel.com/research/mrl/research/opencv/.

[9] A. W. Black and P. A. Taylor, "The festival speech synthesis system: System documentation," Human Communciation Research Centre, University of Edinburgh, Scotland, UK, Tech. Rep., 1997, available at http://www.cstr.ed.ac.uk/projects/festival.html.

[10] X. Huang, F. Alleva, H.-W. Hon, M.-Y. Hwang, and R. Rosenfeld, "The SPHINX-II speech recognition system: an overview," *Computer Speech and Language*, vol. 7, no. 2, pp. 137–148, 1993.

[11] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, 2001.

[12] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[13] J. Serra, *Image analysis and mathematical morphology*. NY: Academic Press, 1982.

[14] M. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Information Theory*, vol. 8, pp. 179–187, 1962.

[15] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 7, 1977.

[16] V. Ostromoukhov, "Digital facial engraving," in *Proceedings of the conference on Computer graphics and interactive techniques*, 1999, pp. 417–424.

[17] M. P. Salisbury, M. T. Wong, J. F. Hughes, and D. H. Salesin, "Orientable textures for image-based pen-and-ink illustration," in *Proceedings of the conference on Computer graphics and interactive techniques*, 1997, pp. 401–406.

[18] P.-M. Jodoin, E. Epstein, M. Granger-Piché, and V. Ostromoukhov, "Hatching by example: a statistical approach," in *Proceedings of the international symposium on Non-photorealistic animation and rendering*, 2002, pp. 29–36.