XXIV IMEKO World Congress "Think Metrology"
August 26 – 29, 2024, Hamburg, Germany

PROGRAMMING AND CONTROL CHALLENGES OF THE PORTRAIT DRAWING PIKTOR-O-BOT

János Csempesz [a], *Tamás Cserteg* [a,b], *Kristóf Abai* [a], *János Nacsa* [a, *]

[a] HUN-REN SZTAKI, 1111, Budapest, Kende u. 13-17, Hungary,
{jcsempesz|csertegt|abai|nacsa}@sztaki.hu
[b] Doctoral School of Informatics, ELTE Eötvös Loránd University, Budapest H-1117, Hungary
* Corresponding author

Abstract – Robotic arms and artificial intelligence-based solutions are all around, but few applications demonstrate the essence of these tools in a way that is easily comprehensible for the general public. Drawing a human face from a photograph seems like a simple task but doing it with an industrial collaborative robotic arm is a complex challenge. While designing the system, an important goal was to incorporate subsystems that mimic completely different, but relatable industrial tasks.

Three different drawing variations are discussed, aligning with the development history of the portrait drawing solution. The first one involves drawing with a marker pen on a white board surface, where one must apply constant force to ensure that the pen touches the surface thus leaves a mark. In the second version, where the drawing is carried out with a so-called brush pen on regular paper, force feedback is not necessary because of the characteristics of the pen. This allows faster drawing for expos and exhibitions with the added benefit of giving visitors a gift to take away. If not only lines, but also regions with a few (4-5) shade intensities are to be rendered by satin finish, the force feedback should be kept within a narrower range. This is the third, graphite pencil utilizing version. Here, the pencil is no longer kept perpendicular to the surface and the sharpening of the pencil is critical.

The paper describes in detail the algorithm of the force-based control for the marker and pencil versions. The calculations and the actual control are executed by a Java-based program on a computer connected to the robot, which has the advantage that it provides a comprehensive set of software tools with which any application can be run on the robot virtually. The design and implementation of this architecture is also outlined in the paper.

For line drawing, the order of the to-be-drawn lines is irrelevant, however it heavily affects the execution time. To overcome this phenomenon, a fast-sequencing algorithm is used for sequencing the lines before the robot is moving to start the drawing process.

Designing the robotic cell for easy transportation generated additional challenges. The paper briefly discusses the cell built up and the steps for putting it into operation after transportation.

***Keywords*:** robot application; image processing; artificial intelligence; portrait drawing; force control

1. INTRODUCTION

There is an ever-increasing expectation in public research institutions that the engineering systems and results, that are produced in research and industrial projects should be presented in a way that is understandable to the wider public. Artificial intelligence (AI), and robotics in particular are areas where it is necessary to provide people real examples for the potentials and limitations of engineering solutions, rather than fictions. This is how robotic portraiture emerged, where the task is clear to all, but the technical solution poses serious challenges. In order not to disconnect the development from the current research and development activities, an additional aspect was the reuse of previous algorithms and softwares (e.g. URMover core, URSztaki_2.0) already proven in industrial projects, which also reduced development time and effort.

The first design was an add-on to an existing robotic cell, where the robot drew on an erasable whiteboard with a marker pen, but due to the wide interest, it soon became clear that a portable design was needed, so it could be taken to various events, and that users wanted to take the drawings with them, so they had to be drawn on paper.

In addition to the detailed design of the AI system that generates the lines from the image, many smaller but essential control and measurement problems had to be solved. The quality of the drawing turned out to be very sensitive to lighting conditions, the correct movement speed of the robot while drawing, the drawing of small radius arcs, the correct choice of the drawing sequence of the necessary lines, the precise calibration, and the stability of the grip of the drawing tool were all tasks that had to be tackled throughout the development.

Many inspiring and exciting ideas came from the visitors, especially from the children, who asked for group drawings, or stood in profile in front of the camera, occasionally even grimacing.

After the design of the mobile system, the idea of creating the portrait by shading instead of line drawing arose, which presented new challenges: the use of pencil instead of pen, the angle of the pencil when drawing, the pressure needed to achieve the shades, what kind of pencil to use on what kind of paper, how to stretch the paper, etc.

1.1. Existing drawing robot experiences

Robotic portraiture has appeared in recent years on a wide variety of platforms and in many different contexts. Technical and artistic aspects must be reconciled with the quality and the speed/efficiency of the drawing, for which a variety of solutions have been developed. The robots used are also diverse, ranging from simple 3-4 DOF devices to industrial robotic arms and even humanoid robots for portrait drawing. There is even a simple commercial device that can be used to draw portraits based on the image taken by the device [1].

Similar to our application, Nasrat et al. [2] and Wang et al. [3] among others were also using industrial robot arms. The first article also shows a wide range of examples of portrait painting by collecting ready-made pictures and drawing times. The summary shows that there are solutions ranging from those that take less than a minute to those that produce a photo-quality black and white image in 17 hours. The second paper is similar to Piktor-o-bot in that it also uses an UR5 robot, but the avatar to be drawn is rather schematic, which is how they achieve fast drawing times.

Shading is a more difficult and less frequently targeted application. An early and well-known example is Paul the robot [4], which was first introduced back in 2011. It draws using a very exciting iterative process, because it occasionally analyses its own work in progress and uses that to determine how to proceed. It doesn't do colouring; it uses a lot of lines to make some areas dark. The robot of Adamik et al. [5] works on a similar principle, using a special pencil gripper and frequent calibration on the fly to create very lifelike portrait using thousands of lines.

The realisation of nuances can only be achieved by controlling the printing force; O'Dowd [6] presents a detailed analysis and a working system, but his system does not focus on expressive portraiture, but on reconstructing an image as accurately as possible with a robot-mounted pencil. He describes many of the difficulties also faced by the Piktor-o-bot.

2. PROBLEM STATEMENT

The motivations of the demonstrator required an easy to setup, easy to use, but versatile robotic application, hence was portrait drawing chosen. It is complicated enough even for not-talented people as well, but intuitive so the subtasks are understandable to everyone.

When a person wants to draw a portrait of themself, the first step is taking a photograph of them. Then, carefully designed image processing algorithms convert this image to a line art-like drawing, that is further processed to a representation that the robot controller can handle. The robot can then draw on a paper or whiteboard surface, depending on the choice of the user. Force control can be utilized if needed and the drawing takes one or two minutes of time, so many people could have access to the demonstration at busy exhibitions.

While developing the demonstrator, many aspects needed careful considerations. Short drawing time was one requirement that needed special attention, including fast image processing as well as optimal ordering of the drawn lines. Not only the drawing needs to be fast, but the development of the application as well, because through many exhibitions lot of feedback was gathered, that was incorporated into the demonstrator. To achieve rapid prototyping regarding the image processing as well as the workflow of the demo, a versatile robotic system is needed including both the software and hardware. As both the calibration and pencil drawing process require accurate force feedback, the inclusion of a force-torque sensor was inevitable. Apart from education, entertainment is also the goal of the demonstrator, and we would not want to displease visitors with unattractive drawings, hence aspects that form the finished look of the drawing needed special attention: adaption of the drawing speed for accurate curves and handling parts of the face with special care. Finally, as the demonstrator gained popularity, it became clear that a portable setup is required which affects both hardware and software design considerations. These unique aspects are detailed in Section 4.

3. SYSTEM ARCHITECTURE

In this section, we describe the hardware and software elements that build up the demonstrator and enable the creation of nice and quick drawings. The drawing process is also detailed, so in later sections the unique characteristics of the demonstrator can be easily understood.

3.1. Key hardware components

A 6-axis robotic arm with a gripper and a force sensor is used with a purpose-built environment. A well renowned robot was selected for the task, a Universal Robots UR5. It is versatile, easy to program and most importantly, can be safely operated around people as it is a collaborative robotic arm. A RobotiQ force-torque sensor is mounted to the robot, so easy calibration procedure can be used and some of the drawing versions require force feedback. A RobotiQ two finger gripper is used for holding the drawing tools, as those range from pencils through brush pens to whiteboard markers. The camera (IDS uEye SX) is mounted on the robot arm as well, so the robot can quickly adapt the photo pose to people with different heights.

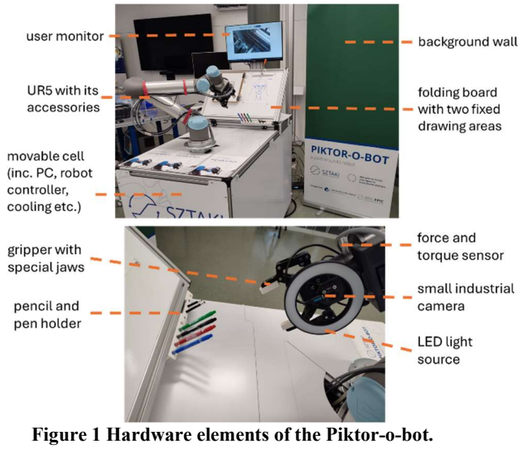

Figure 1 Hardware elements of the Piktor-o-bot.

There are two fixed drawing surfaces, one for the brush pen drawings and the other for the force-controlled pencil drawings. Both surfaces can hold an A4 sized paper. There are fixed slots for the drawing tools next to the drawing surfaces.

The robot is mounted on a movable aluminum frame, that

encloses all computing devices. The drawing surface and the robot can be folded onto the frame, which enables easy transportation. The hardware elements are shown on Figure 1, while the structural design is discussed in Section 5.

3.2. Software architecture

Not only the robot, but also its controller software needs to be versatile. The heart of the process is the URMover framework (see Figure 2), that is a universal purpose robot controller, developed for easy prototyping of high-level manipulation tasks. It provides the functionalities for the Piktor-o-bot core library to interact with the image processing pipeline, the robot controller, the sequence planner, as well as the peripherals (camera, gripper, etc.).

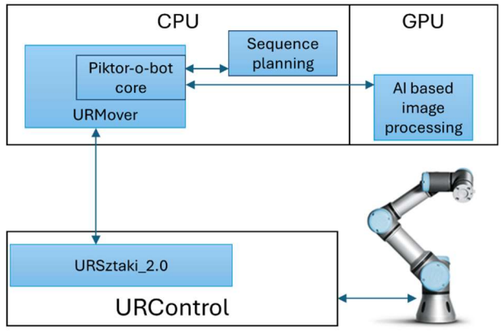

Figure 2 Software architecture of the Piktor-o-bot.

The URSztaki_2.0 middleware layer is a request-response communication service enabling high level programming languages to command and monitor the robot. The server component of this layer is implemented in URScript language and runs on the robot controller. The client (proxy) library of this layer is implemented in Java-SE language. The middleware layer can run complex commands on the robot in a sequential synchronized way or move the robot along an online on-the-fly computed trajectory.

The goal of the AI-based image processing pipeline is to generate a set of lines from the image taken by the camera, while keeping the characteristics of the person photographed. Its input is the colour image, and the output is a binary image with a set of lines. A vectorization step is also required to translate the lines for the robot. Section 4.1 describes the pipeline in more detail, while Section 4.2 discusses the problem of line ordering with the goal of minimizing the drawing time.

The subsystems are managed by the Piktor-o-bot core application, that controls the whole demonstration process.

3.3. Drawing process

The pictures are taken by the robot moving into a predefined photo pose, while the human stands in front of the camera. The robot finds the optimal height by moving the camera up and down. A real-time face detector [7] is used to adjust the height of the camera, achieving a centred face position on the image taken. After taking the image, the AI-based image processing algorithms produce the to-be-drawn lines and the drawing itself starts.

First that drawing tool is picked up, then the continuous lines are drawn on the surface. In between lines, the tool is lifted away from the surface by a few millimetres, then the next line is drawn. After finishing the last line, the drawing tool is put back to its slot.

4. UNIQUE CHALLENGES / IMPLEMENTATION DETAILS

In this section, we discuss the unique implementation details of the application, that enabled us to solve the arising problems discussed in Section 2.

4.1. Image processing

As mentioned earlier, an appealing final drawing is one of the main goals of this demonstrator. This requires a carefully crafted image processing pipeline, which is described more in detail in [8]. For summary, a brief description of the steps is given and shown on Figure 3 [9].

For the first step, the image is cropped to contain only the face (or faces in case of group drawings), and the background is removed. Then, edges are detected using a neural network (DexiNed [10]). This gives the core of the final image.

Showcasing the demonstrator at several exhibitions and meeting unexpected poses, facial expressions, etc. we realized that few parts of the face need special care. A segmentation is needed for handling these special cases, which is solved by RTNet [11]. The following parts of the face are handled separately: the teeth are removed, as we experienced that even nice smiles became horrifying after drawing and only the outline of the hair and the eyebrows are drawn. The eyes are considered with more details to create a more lifelike end result. The iris and the pupil are detected using the MediaPipe framework [12] and added to the to-be-drawn image. These parts of the image processing pipeline required significant computing power to run quickly, therefore a GPU was installed in the controller PC.

The results of the segmented parts and the full face are merged in one image, then skeletonization is carried out to facilitate the vectorization step, which is carried out by the autotrace library [13]. The output of this last step is a set of

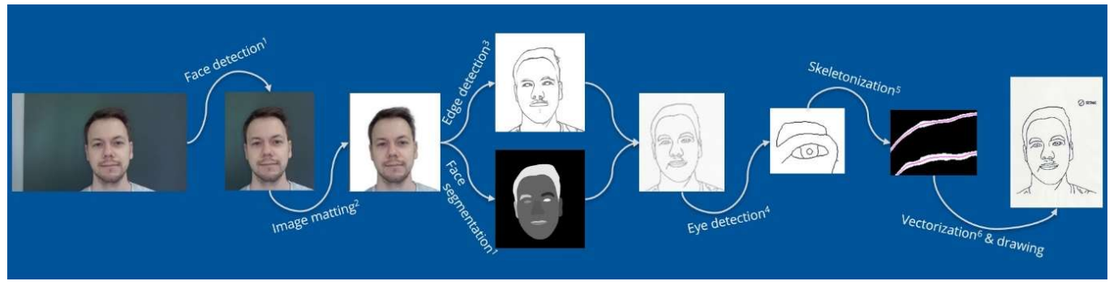

Figure 3 Image processing pipeline of the Piktor-o-bot [9].

lines represented by cubic curves, that are tessellated into high fidelity polygons using the Bézier algorithm.

4.2. Sequence planning

The drawing process must be as fast as possible so the trajectory must be optimized for time. The model chosen for the trajectory is the set of polygons.

For the optimization model the final timed trajectory in 3D space could also be chosen taking account the robot dynamics and capabilities (speed, acceleration, etc.), but that would have been a much more complex problem and fortunately the dynamics are negligible. Similarly to a planned optimal route for a vehicle to go from point A to point B, only the topology of the roads matters, not the dynamics of the vehicle.

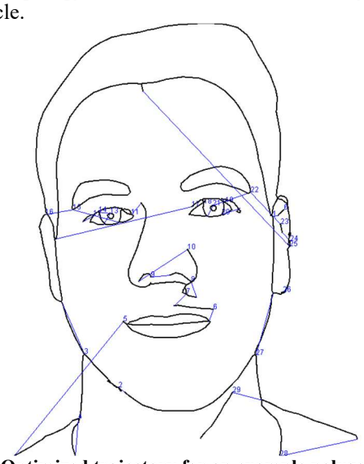

Figure 4 Optimized trajectory for an example polygon set (black lines mark the to-be-drawn lines, and blue lines the in-air-moves between them).

Drawing a single continuous line always takes the same amount of time according to that line, so what really matters is in which order the lines are drawn and in which direction each line is drawn (reversed or not). Basically, what determines the final drawing time is the time moving in the air with raised pen between the lines.

To calculate the optimal trajectory, the problem can be formulated as a General Traveling Salesperson Problem (GTSP). Solvers are readily available for this kind of task, and the ProSeqqo solver [14] was chosen.

When constructing the optimization problem, the actual speed characteristics along a line are discarded and constant speed is used. The cost of lifting and putting down a pen between two drawn lines is also incorporated into the model. The actual parameters for the selected robot speeds were specified through experiments on the robot.

The ProSeqqo solver is called only when the drawing is initiated, running parallel with the robot picking up the pen from its slot and moving in front of the drawing area. The solver usually finishes before the robot finishing its movements, so it is quite a streamlined experience from the users' perspective. An example sequence is shown on Figure 4, where black lines mark the to-be-drawn lines, and blue lines the in-air-moves between them. More details one the problem representation and experiments are available in the paper [14].

4.3. Adaptive speed control over the trajectory

A well-designed plan still requires excellent realization; therefore, the trajectory needs to be carefully planned. The robot needs to follow the generated lines as close as possible, which requires that the robot's dynamic behaviour is controlled while the trajectory is planned.

As described earlier, the drawing is represented as a list of 2D polygons. To draw a line in 3D space the lines are projected to the 3D drawing board becoming 3D polygons. Before drawing, a trajectory is computed for the robot to go over this polygon with a given maximum speed and given acceleration/deceleration which come from the dynamics of the robot. The UR5 robot is controlled at a 125Hz frequency, which means that a new target position needs to be fed at every 8th millisecond.

The accuracy of the final drawing is depending on the curvature and the speed of the robot at given point on the polygon. Hence, the final trajectory takes time into account and has variable speed. The trajectory of a continuous curve starts and ends with zero speed and in every timestamp the speed must match the curvature of the polygon.

This is an important calculation that determines the accuracy of the final drawing and the drawing time. In the trajectory, it is necessary to decelerate in time before approaching sharp turns, based on actual speed, deceleration, distance to the turn and the curvature. Otherwise, the robot can accelerate to maximum speed.

4.4. Force-controlled shaded drawing

The versatility of robotic solutions can be presented very well with force-feedback applications. Therefore, a shaded, pencil drawing mode is developed. This mode fills regions

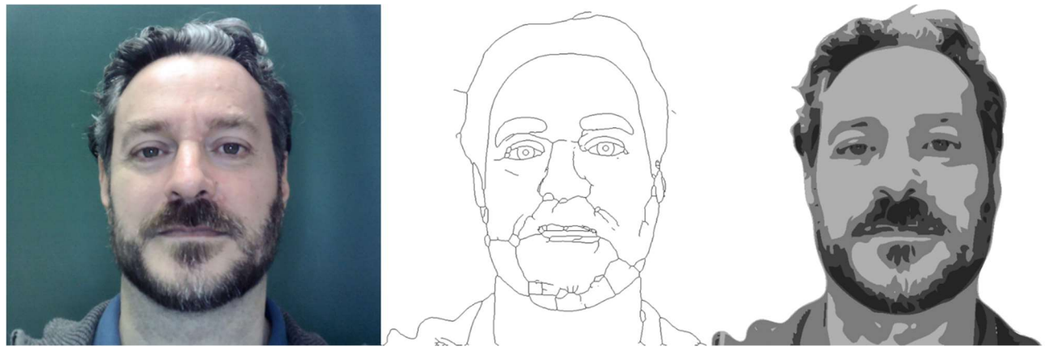

Figure 5 An image with the processed result for brush pen and shaded drawing.

with pencils instead of drawing lines. As a first step, grayscale drawings are considered, which means that the “colour”, the intensity is defined by the pressure with which the pencil is pressed to the paper. Thus, a different image processing method is required to generate the regions as well as force control is required to create the different intensity levels.

The force control is based on the force-torque sensor measurements. The force values are continuously read by the robot and sent to the high-level controller through the middleware. The processing of the force signals and the actions based on them are done in the high-level controller java thread. The paradigm of the force control is to maintain a target force perpendicular to a spatial plane. In the case of drawing this plane is the drawing surface. The force control is implemented as a tunable PID controller thread that modifies the online trajectory based on the force vector awakening at the tip of the pencil. The objective is to maintain a constant force value in the direction perpendicular to the drawing surface.

The shaded drawing mode required a separate image processing method. With our hardware setup, four gray intensities can be robustly created, and four shades seem to be enough for humans to enjoy a grayscale drawing. The image processing in this case is a colour reduction type vectorization preceded by background removal. Background is always matched with the whiteness of the paper. The output is generated with the autotrace tool, and it is a set of closed polygons, each with a gray intensity.

The gray intensities are converted to target force values. Intuitively darker gray is matched with larger target force. The lower intensities are drawn with a type H pencil tilted with 60 degrees with force 2N-8N. Higher intensities are drawn with type B pencil tilted with zero degrees (held perpendicular to the surface) with force 8N-16N. The reason is that type H pencil can shade paler shades, the tilting can cover larger area meaning quicker shading speed, but the pencil tip may break at greater forces. Type B pencil has softer tip resulting in stronger shades and covering greater area and it can tolerate larger forces when held perpendicular. A type B pencil pushed with 16N force leaves a near black shade on paper.

For generating the robot trajectory, the polygon areas are filled with zig-zag pattern lines. Each line is matched the target force of the corresponding region. The drawing is organized in a way, that the robot begins with the lower intensity regions, then switches pencil and finishes with the darker regions.

An example portrait is shown on Figure 5 with the taken image, and the processed results for the brush pen drawing and the shaded pencil drawing versions.

5. TRANSPORTATION CHALLENGES

As the demonstrator gainer popularity, it became clear, that frequent transportation of the robotic cell requires attention from hardware and software aspects as well. An easily transportable, but rigid structure is needed. From the design point of view, it is easier to calibrate the drawing surface at each setup, rather than constructing a too precise cell setup. The robot is equipped with a force-torque sensor anyways; thus, the only requirement is that the calibration needs to be quick and easy to do.

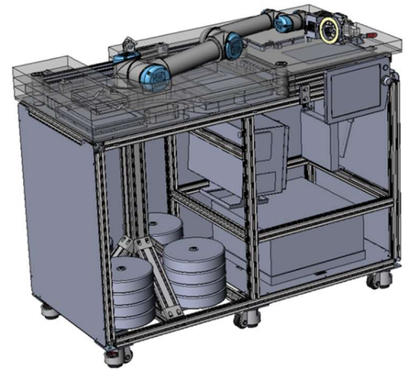

Figure 6 CAD model of the demonstrator in folded state for transportation.

5.1. Hardware solutions

The physical robotic cell is constructed from aluminum profiles (see Figure 6). On top of the frame, there is a drawing board that provides two fixed drawing areas: one for drawing with low force and another for pencil drawings with high force. Both surfaces are suitable for A4-sized paper. The drawing surface designed for high-force applications securely holds the paper with four strong clamps, while the low-force side employs easily detachable linear jaw clamps for simple paper replacement. Attached to this drawing board is a fixture for holding pencils and pens, from which the robot can pick up the desired drawing tool.

Great emphasis was placed on ensuring maximum rigidity of the aluminum frame structure to avoid vibration and undesired effects from frame movement or deformation. However, the mass of the cell frame alone does not provide sufficient stability, thus additional weights need to be placed at the bottom of the robot mounting bracket. This does not compromise the portability and ease of movement as the weights are removable for transportation. The structure can be moved on the six lockable rubber wheels located at the base of the unit. The cell enclosure is made of metal-coated sandwich panels. Inside the frame, all necessary equipment for operating the cell is housed, including the robot controller, PC, networking equipment, cooling system, and the weights. For transportation, the drawing board can be folded down, the monitor mounting profile can be dismantled, and all necessary accessories such as pens, pencils, papers, mouse, keyboard, screws, fasteners, and fixtures can be packed into a custom EPE (expanded polyethylene) foam system. The entire packed cell also comes with a durable and waterproof external cover to protect it during transportation.

5.2. Software solutions

With the frequent disassembly and assembly of the demonstrator, it is not possible to achieve the accuracy required by the robot for the drawing area setup (or only with a budget that cannot be justified). Therefore, an easy-to-use calibration is developed.

The calibration consists of recording the marker, pencil pickup poses, tactile scanning the drawing areas, determining the hotspot offset poses of the marker/pencil tips relative to the robot effector coordinate system grasping the marker/pen, recording the photo pose, and allowed vertical photo pose

range, and recording joint waypoints moving between photo poses and drawing poses to prevent invalid robot movements.

The main Java program is also a robot teaching framework (the drawing module is basically a submodule of the framework), this way the calibration is an integrated wizard-like process, using the moving, touching, pose managing-computing functions of the framework.

6. EXPERIENCES AND CONCLUSIONS

The Piktor-o-bot demonstrator has been showcased at 60+ events (a third of them by moving off-site to national and international exhibitions and expos in the last years, including the 2nd Stuttgart Science Festival.) The design choices that have been made in the construction of the demonstrator have resulted in a device that is relatively easy to set up. From the moment of arrival until the first portrait is drawn, the setup can be completed in under an hour, and the disassembly takes similar amount of time. The biggest advantage of the flexible design is that it was easy to adapt to the different conditions at each location. We were able to adapt to the local lighting conditions by being able to choose different photo positions at the different locations.

The demonstrator gained large attention and sparked numerous discussions on the various aspects of automation, robotics, and artificial intelligence. During these events we found that children were more likely to take the initiative in requesting a portrait. Adults, on the other hand, were more inclined to wait for someone else to be drawn, but then asked more questions and initiated more conversations. In most cases these, questions were directed at the applications for this demonstrator, which often resulted in disputing the underlying technologies. These discussions demonstrate that the primary objectives of education and the generation of discourse surrounding the Piktor-o-bot demonstrator are continuously achieved at public appearances.

ACKNOWLEDGMENTS

Acknowledgement shall be expressed to the International Measurement Confederation (IMEKO), especially to the organizers, of the XXIV. IMEKO World Congress for cooperating in the demonstration of the Piktor-o-bot robot at the IMEKO World Congress in Hamburg, Germany, 26-29 Augustus 2024.

FUNDING STATEMENT

This work was supported by European Union within the framework of the National Laboratory for Artificial Intelligence (RRF-2.3.1-21-2022-00004) and by the EPIC Centre of Excellence in Production Informatics and Control Horizon 2020 research and innovation programme under grant agreement No 739592

REFERENCES

[1] ‘Draw Me Bot: The AI-Based Drawing Robot Photo Booth’. Foto Master. Accessed: Apr. 10, 2024. [Online]. Available: https://fotomaster.com/products/draw-me-bot/

[2] S. Nasrat, T. Kang, J. Park, J. Kim, and S.-J. Yi, ‘Artistic Robotic Arm: Drawing Portraits on Physical Canvas under 80 Seconds’, *Sensors*, vol. 23, no. 12, Art. no. 12, Jan. 2023, doi: 10.3390/s23125589.

[3] T. Wang *et al.*, ‘RoboCoDraw: Robotic Avatar Drawing with GAN-Based Style Transfer and Time-Efficient Path Optimization’, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 06, Art. no. 06, Apr. 2020, doi: 10.1609/aaai.v34i06.6609.

[4] P. Tresset and F. Fol Leymarie, ‘Portrait drawing by Paul the robot’, *Computers & Graphics*, vol. 37, no. 5, pp. 348–363, Aug. 2013, doi: 10.1016/j.cag.2013.01.012.

[5] M. Adamik, J. Goga, J. Pavlovicova, A. Babinec, and I. Sekaj, ‘Fast robotic pencil drawing based on image evolution by means of genetic algorithm’, *Robotics and Autonomous Systems*, vol. 148, p. 103912, Feb. 2022, doi: 10.1016/j.robot.2021.103912.

[6] P. O’Dowd, ‘A Robot That Draws and Shades with Tactile Force Feedback Sensed Through a Pencil’, presented at the Proceedings of EVA London 2019, BCS Learning & Development, Jul. 2019. doi: 10.14236/ewic/EVA2019.19.

[7] G. Bradski, ‘The OpenCV library’, *Dr. Dobb’s Journal of Software Tools*, 2000.

[8] A. T. Hoang, J. Csempesz, T. Cserteg, and Zs. J. Viharos, ‘PIKTOR-O-BOT: Integrated image processing algorithms for portrait drawing robot applications’, presented at the XXIV IMEKO World Congress, ‘Think Metrology’, Hamburg, Germany, Aug. 2024.In press.

[9] T. Cserteg, A. T. Hoang, K. Kis, J. Csempesz, and Z. J. Viharos, ‘Piktor-O-bot: The robotic face-Drawing solution’, *ERCIM NEWS*, no. 132. EUROPEAN RESEARCH CONSORTIUM INFORMATICS & MATHEMATICS, 2004, ROUTE LUCIOLES, BP 93, SOPHIA ANTIPOLIS CEDEX, 06902, FRANCE, pp. 24–25, Jan. 2023.

[10] X. Soria, A. Sappa, P. Humanante, and A. Akbarinia, ‘Dense Extreme Inception Network for Edge Detection’, *Pattern Recognition*, p. 109461, Feb. 2023, doi: 10.1016/j.patcog.2023.109461.

[11] Y. Lin, J. Shen, Y. Wang, and M. Pantic, ‘RoI Tanh-polar transformer network for face parsing in the wild’, *Image and Vision Computing*, vol. 112, p. 104190, Aug. 2021, doi: 10.1016/j.imavis.2021.104190.

[12] C. Lugaresi *et al.*, ‘MediaPipe: A Framework for Perceiving and Processing Reality’.

[13] W. Martin, ‘autotrace’. Accessed: Apr. 01, 2024. [Online]. Available: https://github.com/autotrace/autotrace

[14] L. Zahorán and A. Kovács, ‘ProSeqqo: A generic solver for process planning and sequencing in industrial robotics’, *Robotics and Computer-Integrated Manufacturing*, vol. 78, p. 102387, Dec. 2022, doi: 10.1016/j.rcim.2022.102387.