Compact, Accurate and Low-cost Hand Tracking System based on LEAP Motion Controllers and Raspberry Pi

Giuseppe Placidi[1][a], Alessandro Di Matteo[1], Filippo Mignosi[2][b], Matteo Polsinelli[1][c] and Matteo Spezialetti[2][d]

[1]*A^{2}VI-Lab, c/o Dept. MeSVA, University of L'Aquila, Via Vetoio Coppito, 67100 L'Aquila, Italy*
[2]*Dept. DISIM, University of L'Aquila, Via Vetoio Coppito, 67100 L'Aquila, Italy*

Keywords: Hand Tracking, Virtual Glove, Raspberry Pi, Occlusions, Remote Operating.

Abstract: The large diffusion of low-cost computer vision (CV) hand tracking sensors used for hand gesture recognition, has allowed the development of precise and low cost touchless tracking systems. The main problem with CV solutions is how to cope with occlusions, very frequent when the hand has to grasp a tool, and self-occlusions occurring when some joint obscures some other. In most cases occlusions are solved by using synchronized multiple stereo sensors. Virtual Glove (VG) is one of the CV-based systems that uses two orthogonal LEAP sensors integrated into a single system. The VG system is driven by a Personal Computer in which both a master operating system (OS) and a virtual machine have to be installed in order to drive the two sensors (just one sensor at a time can be driven by a single OS instance). This is a strong limitation because VG has to run on a powerful PC, thus resulting in a not properly low-cost and portable solution. We propose a VG architecture based on three Raspberry Pi (RP), each consisting of a cheap single board computer with Linux OS. The proposed architecture assigns an RPi to each LEAP and a third RP to collect data from the other two. The third RP merges, in real time, data into a single hand model and makes it available, through an API, to be rendered in a web application or inside a Virtual Reality (VR) interface. The detailed design is proposed, the architecture is implemented and experimental benchmark measurements, demonstrating the RPi-based VG real-time behaviour while containing costs and power consumption, are presented and discussed. The proposed architecture could open the way to develop modular hand tracking systems based on more than two LEAPs, each associated to one RP, in order to further improve robustness.

1 INTRODUCTION

Computer vision (CV) (Voulodimos et al., 2018) is actually applied with success for the development of touchless hand tracking systems due to the low cost, increasing precision, fastness and high versatility in recognizing every hand size and silhouette (Oudah et al., 2020). On the contrary, wearable gloves (WG) use devices installed directly on the hand and fingers (Battaglia et al., 2015; Luzhnica et al., 2016; Wang et al., 2020) which make them haptic and precise. However, WG expensive, need to be specifically designed for a hand size and shape and could be very limiting for the movements. CV-based approaches (Placidi, 2007; Placidi et al., 2013; Erden and Cetin, 2014; Marin et al., 2016; Placidi et al., 2017; Placidi et al., 2018; Kiselev et al., 2019; Shen et al., 2019; Yang et al., 2020; Ameur et al., 2020; Placidi et al., 2021) use the interpretation of video-collecting devices, usually sensors operating also in the visible or in the infrared (IR) range, placed at a certain distance from the hand. The key advantage of CV-based systems is that no physical contact is required and the movements are free, fluid and natural, being the hand unforced to wear anything, and it could naturally grip specialized tools to carry on specific procedures. For these reasons, CV hand tracking systems are growing especially in human-system interaction, to improve the communication between users and computers, virtual reality environments, devices and robots, also from remote, to perform medical procedures and for rehabilitation (Ankit et al., 2011; Avola et al., 2013; Placidi et al., 2015; Imran and Raman, 2020;

[a] https://orcid.org/0000-0002-4790-4029
[b] https://orcid.org/0000-0001-9599-5730
[c] https://orcid.org/0000-0002-4215-2630
[d] https://orcid.org/0000-0001-5786-3999

Placidi, G., Di Matteo, A., Mignosi, F., Polsinelli, M. and Spezialetti, M.
Compact, Accurate and Low-cost Hand Tracking System based on LEAP Motion Controllers and Raspberry Pi.
DOI: 10.5220/0010880900003122
In *Proceedings of the 11th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2022)*, pages 652-659
ISBN: 978-989-758-549-4; ISSN: 2184-4313

Mahdikhanlou and Ebrahimnezhad, 2020; Chen et al., 2015; Jin et al., 2016; Erden and Cetin, 2014; Liu and Zhang, 2014; Carrieri et al., 2016; Zhang et al., 2019; Petracca et al., 2015; Moro et al., 2016).

Among the CV based gloves, those implemented with the LEAP (LEAP[1]) sensor are the most diffuse (Placidi et al., 2017; Placidi et al., 2018; Kiselev et al., 2019; Shen et al., 2019; Yang et al., 2020; Ameur et al., 2020) due to the fact that LEAP is accurate, low-cost, compact (Bachmann et al., 2014) and can effectively be used for hand tracking.

LEAP sensor uses 3 IR light sources and two detectors to obtain 3D visual information saved and reproduced almost simultaneously (more than 60fps) from the server. One of its advantages is that it is appropriate for different hand sizes (adults and children), as well as for different hand shapes (healthy people and patients with residual infirmities).

However, all CV hand tracking systems, including LEAP-based, have to face the problem of occlusions, occurring when some hand joints are invisible to the sensors, being occlusions either produced by external objects grasped by the hand or by parts of the hand itself (self-occlusions).

Due to occlusions, CV-based tracking systems can fail to correctly reproduce position/trajectory of the hand or parts of it because the position of the invisible parts are guessed, thus resulting in inaccurate and unstable representations. This could be negligible when just raw gestures need to be reproduced, but crucial when finer movements are used in tele-operated applications, such as tele-surgery or operations in dangerous environments (Choi et al., 2018; Mueller et al., 2019; Smith et al., 2020).

The only way to reduce the impact of occlusions is to use multiple sensors surrounding the scene, in a multiple-view arrangement, all sensors synchronized each-other and all contributing to recover the hand model in real-time (occlusions are solved because points which are invisible to one sensor can probably be visible to another).

Recently, several works have been published with the aim of improving hand tracking accuracy by combining LEAP data with those of other devices or data from multiple LEAPs (Marin et al., 2016; Mahdikhanlou and Ebrahimnezhad, 2020; Yang et al., 2020; Kiselev et al., 2019; Shen et al., 2019; Placidi et al., 2017; Placidi et al., 2018; Placidi et al., 2021).

In particular, in (Marin et al., 2016) a LEAP is supported by a Depth camera. The system has very good accuracy (regarding gesture recognition) but low frame rate (15fps) making it not suitable for applications that requires higher frequency (30fps or greater) to track natural hand movements. Moreover, due to the occlusions between fingers (self-occlusions), the method performs well only when the hand is in ideal orientations/positions.

In (Mahdikhanlou and Ebrahimnezhad, 2020) a LEAP is supported by an RGB webcam to improve the quality of the recognition of symbols in the 3D American sign language datasets. Aim of the proposed system is to reduce the ambiguities in gesture recognition: the RGB webcam is used as an auxiliary system, being it unable to furnish specific spatial information. The same gesture recognition problem for identification of American sign language and HandicraftGesture is solved accurately with just one LEAP (Yang et al., 2020).

Kiselev et al. (Kiselev et al., 2019) use three LEAPs for gesture recognition. The Authors show that by increasing the number of sensors, the accuracy also increases due the fact that the number of occlusions decreases. Moreover, the use of multiple sensors of the same type greatly improves the performance of the data integration strategy due to the easiness in comparing similar models. However, since just one LEAP at time can be driven by a single operating system instance, the used client/server architecture described in the paper suggests that at least three different computers have been used (expensive and critical in terms of synchronization). In addition, as two of the three LEAPs are coplanar, they mostly contribute to increase the active region but have low influence in reducing occlusions. Finally, the performance of the system, in terms of frame rate, has not been discussed.

Shen et al. (Shen et al., 2019), solve the problem of occlusions in gesture recognition by proposing the use of three LEAPs placed with their long axes on the medium points of the sides of an equilateral triangle. Though the paper deeply discusses on the system assembly, calibration, data-fusion and results in terms of position/orientation accuracy, no mention is dedicated to the resulting efficiency of the system in terms of fps.

Virtual Glove (VG) (Placidi et al., 2017; Placidi et al., 2018; Placidi et al., 2021) is a system based on the synchronized use of two orthogonal LEAPs (Fig. 1) for reducing the probability of occlusions. In particular, in (Placidi et al., 2017), VG was first presented in a raw assembly; in (Placidi et al., 2018) it was refined and calibrated and, finally, in (Placidi et al., 2021) a new strategy for real time data integration from both sensors was presented and discussed.

Better results regarding occlusions reduction could have been obtained with three LEAPs on a equilateral configuration, as in (Shen et al., 2019), but

[1]https://www.ultraleap.com/

with serious problems with the real-time maintenance (at least 30fps) on a low-cost computer. Though, in principle, the paradigm of VG (Placidi et al., 2017) is applicable to any number of sensors placed in any angular configuration, the choice of two orthogonal sensors represents a good compromise between optimization/positioning of the region of interest (ROI), precision and efficiency. In fact, position/dimensions of the ROI and precision with respect to the angle are leaved to project-related considerations: what is critical in VG is the possibility of maintaining the real time (at least 30 fps are necessary for hand motion fluidity).

Since now, also in its final ensemble (Placidi et al., 2021), VG was limited to the use of just two LEAP sensors because of maintaining efficiency in driving the LEAP sensors with a single powerful personal computer.

Aim of the present paper is to design and test a different hardware architecture to drive LEAP sensors in VG in order to allow, in the future, the extension of its concept to more than 2 sensors while maintaining both cost and power consumption low.

The rest of the manuscript is structured as follows: Section 2 reviews VG assembly. Section 3 details the proposed architecture. Section 4 presents experimental benchmark data and discussion. Finally, Section 5 concludes the manuscript and delineates future developments.

2 ORIGINAL VG CONFIGURATION

As discussed above, and referenced in (Placidi et al., 2018), VG is composed by two orthogonal LEAP sensors each lodged on a side of a square-angle aluminium support of side 25 cm (see Figure 1). The centre of each LEAP was positioned at 18.5cm from the internal part of the corner of the support: this was established to optimize the signal in a cylindrical region of interest (ROI) of radius 10cm and height 21cm while also reducing the effects of IR interference between sensors. The two sensors were calibrated to allow data integration between them and to construct a single hand model with data coming from both sensors in real time (about 60 fps). Different fusion strategies were presented: the original, simple, used data from just one sensor at a time, those having the most favourable view of the hand, in mutual exclusion (Placidi et al., 2018); the second, smarter than the first (but slower), used data from both sensors in each frame and merged the joints from one sensor with those from the other according to the calculation of temporal behaviour (temporal smoothness) of each joint (Placidi et al., 2021).

In either cases, data from the sensors had to be collected by different machines. In fact, a limitation of the LEAP sensor is that an instance of the Operating System (OS) can drive no more than one sensor. The simultaneous management of multiple LEAP sensors was solved by using a virtual machine. The virtual machine (Slave) was installed on the physical machine (Master): one sensor was assigned to Master and the other to Slave, thus allowing to each machine to instantiate its own, single, driver. Data provided by both machines were rerouted towards a server (hosted on the Master) that provided to send data of both devices to the Master machine.

The server driven data from both LEAP sensors and provided data fusion: data fusion strategy is also implemented on the Master OS.

The previous stratagem allowed to drive 2 LEAP sensors in a single machine: however, the host computer was required to be powerful. Another choice would be to use different PCs for different sensors, but this would make VG architecture even more complex, expensive and cumbersome than the one used, at least until now.

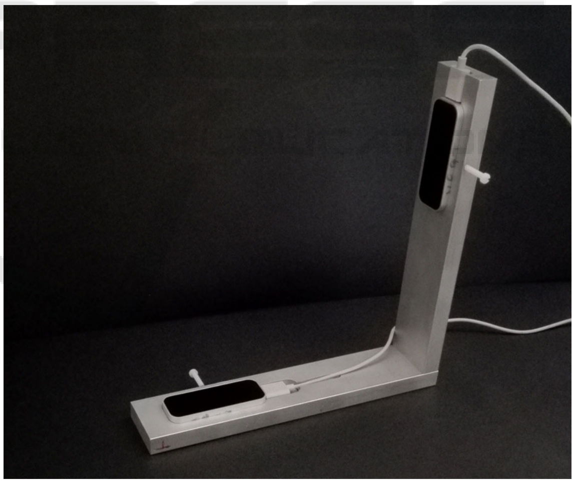


Figure 1: Orthogonal LEAP sensors assembly in VG.

3 RPi-BASED VG CONFIGURATION

Recent advances in the field of microprocessors has allowed the development of cheap, compact and powerful computers to be used in several applications. Raspberry Pi (RP) 4 model B (https://www.raspberrypi.org/) is a single board computer equipped with a 64 bits quad-core ARM8 Broadcom BCM2711

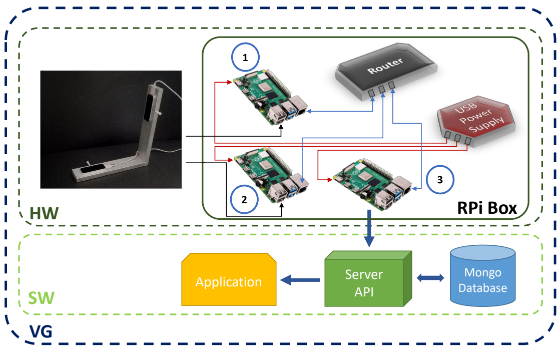


Figure 2: VG architecture based on RPi: each LEAP sensor is associated to one RPi (1 and 2) and data from hand models of both sensors are passed RP3. On each RPi an instance of Linux OS runs. Besides OS, RP1 and RP2 just collect data from the respective sensor; RP3 executes the software to merge the two hand models into a single one (to resolve the occlusions) and the graphic interface for the user interaction.

1.5 Ghz processor, 4GB LPDDR4 Ram and an extensible MicroSD. The operating system is derived from Linux and it is called Raspberry Pi OS. The cost of each RP is about 60 USD and the dimensions of the board are 60mm x 90mm. There also exist more recent and powerful versions but the previous one is that we have integrated into our project. The new hardware architecture of VG is based on three RPi (i=1:3), as reported in Figure 2. Instead of using a single powerful PC, we have used three RP: two assigned to drive the LEAP sensors and the other, in cascade, to collect data from the others, to perform data fusion in a single hand model and to run the virtual reality environment. The RPs are interconnected through a small router using Ethernet cables. Data from the RP1 and RP2 are collected into the RP3 by using a web socket communication protocol.

Since the LEAP driver is available just for X86 processors, we have used the emulator Box86 to emulate X86 instructions into an ARM processor. In fact, it allows to execute Linux x86 applications on Linux not X86, as the ARM case. In order to run Box86, the OS has to be 32-bits. Further, Box86 uses native versions of some OS libraries, thus ensuring good performance. Box86 integrates the dynamic re-compilation (Dynarec) for the ARM platforms: in this way the execution time is from 5 to 10 time faster than using an interpreter. Box86 source code is released by GitHub (https://github.com/ptitSeb/box86).

The other two processes necessary to be run on the RPi are Leapd and Visualizer.

Leapd is a daemon process responsible for the creation of the numerical hand model to be made available for the other processes. A connection to Leapd is possible either from a proprietary SDK or from a websocket client. In fact, the daemon hosts a websocket server. Leapd allows the hand model formation and its transfer outside the board (we use websocket to transfer it to RP3).

Visualizer is an application, furnished by the SDK, which allows to visualize the hand model generated from the LEAP sensor. The final ensemble of the RPi-box is reported in Figure 3.

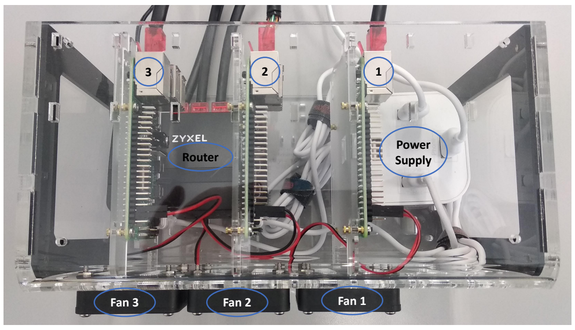


Figure 3: Upper side view of the RPi-box: it hosts the three RPi (1, 2 and 3), a multiple USB power supply on the right (to reduce the number of wires exiting from the box) and a modem to switch data between RPi (left side, on the floor). The cooling system is composed by three fans (black components in the bottom). The transparent box is specifically produced to host a maximum of 4 RPi.

Each RPi is equipped wit a 64 GB Sandisk SD card, transfer rate of 120 MB/sec.

The installation procedure starts by compiling Box86 on the RPi s(in the proprietary OS of RPi, the installation requires the use of the apt package manager, being the command ”cmake” absent). In order to save the execution file of Box86 in the OS, the command ”make install” has to be executed followed by a restart of the system.

Once Box86 is installed, it is possible to execute the deamon of the LEAP sensor first by downloading the file ”LeapDeveloperkit” for Linux from the proprietary site (https://developer.leapmotion.com/sdk-leap-motion-controller). The executable file has to be in the path x86/usr/sbin: its execution is possible by running the command ”sudo box86 leapd”.

A similar procedure needs to be executed to install the process Visualizer. However, it requires the Qt library: for this reason, Qt has to be installed on the OS before running the Visualizer. After that, the command ”sudo box86 Visualizer” is sufficient to run the Visualizer (as shown in Figure 5).

Finally, the client/server software developed in (Placidi et al., 2018) with the fusion strategy presented in (Placidi et al., 2021) has been tested on the new architecture. As discussed above, the web-based

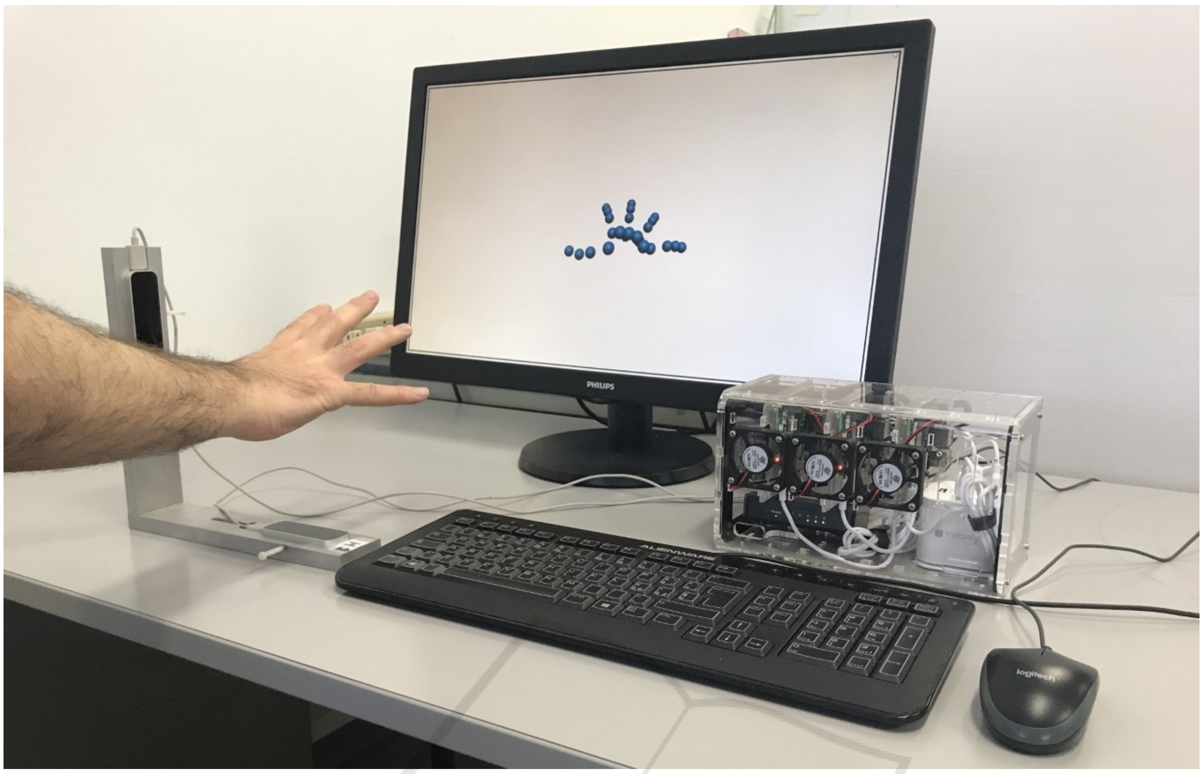
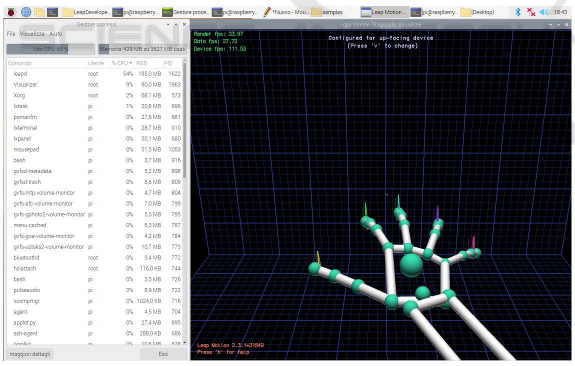


Figure 4: RPi-based VG in its final embodiment, composed by: RPi-box (lateral view, the cooling system is now visible), sensor support, wide-screen, keyboard and mouse.



Uso CPU: 69 %
Memoria: 429 MB su 3827 MB usati

Comando	Utente	% CPU	RSS	PID
leapd	root	54%	185,0 MB	1622
Visualizer	root	9%	90,0 MB	1963
Xorg	root	2%	68,1 MB	573
lxtask	pi	1%	20,8 MB	996
pcmanfm	pi	0%	27,6 MB	681
lxterminal	pi	0%	28,7 MB	910
lxpanel	pi	0%	30,1 MB	680
mousepad	pi	0%	31,5 MB	1053
bash	pi	0%	3,7 MB	916
gvfsd-metadata	pi	0%	5,2 MB	898
gvfsd-trash	pi	0%	8,6 MB	809
gvfs-mtp-volume-monitor	pi	0%	4,7 MB	804
gvfs-afc-volume-monitor	pi	0%	7,0 MB	799
gvfs-gphoto2-volume-monitor	pi	0%	5,0 MB	795
menu-cached	pi	0%	6,3 MB	787
gvfs-goa-volume-monitor	pi	0%	4,2 MB	784
gvfs-udisks2-volume-monitor	pi	0%	10,7 MB	775
bluetoothd	root	0%	3,4 MB	772
hciattach	root	0%	116,0 KB	744
bash	pi	0%	3,5 MB	726
pulseaudio	pi	0%	8,9 MB	722
xcompmgr	pi	0%	1024,0 KB	716
agent	pi	0%	4,5 MB	704
applet.py	pi	0%	27,4 MB	695
ssh-agent	pi	0%	288,0 KB	686
lxpolkit	pi	0%	10,6 MB	678

```
Render fps: 23.97
Data fps: 27.72
Device fps: 111.53
Configured for up-facing device
(Press 'v' to change)
Leap Motion 2.3.1+31549
Press 'h' for help
```

Figure 5: A screenshot collected from the RP1. The system resources are shown alongside the Leap Visualizer which renders a stick model of the hand. In order to represent the hand movement, colored lines at the tips of the fingers were added. This negatively affects the Frame Rate.

rendering software is only necessary on RP3. In fact, the data models from RP1 and RP2 have just to be passed to RP3 for model fusion and visualization in the virtual environment.

4 RESULTS

The first test was conducted on RP1 and RP2. The experiment consisted on tracking a free moving hand for a total of 10 minutes. Since RP1 and RP2 are twins with respect to the hardware and software they use, the results are very similar and, for this reason, reported only for RP1, in Figure 6. Data in Figure 6 show that RP1 and RP2 are capable to run the whole Leap Motion Controller software, including the Leap Visualizer, at a frame rate which is about 33 fps. In this configuration, the CPU is not completely used (in the average, just 63% of the CPU is used). For our purposes, a first for of optimization can be obtained by excluding the Leap Visualizer from RP1 and RP2 (the model is visualized as a result of RP3). In this case, the resulting frame rate of both RP1 and RP2 is increased to about 37 fps and the CPU usage is reduced to 57%, in the average. The reduced CPU usage suggests that further optimization is possible to boost the frame rate.

The final test was conducted on RP3 in the final assembly for a free hand moving experiment of the same type, and the same duration, of that used for RP1 and RP2. Figure 7 shows that RP3 is capable to ren-

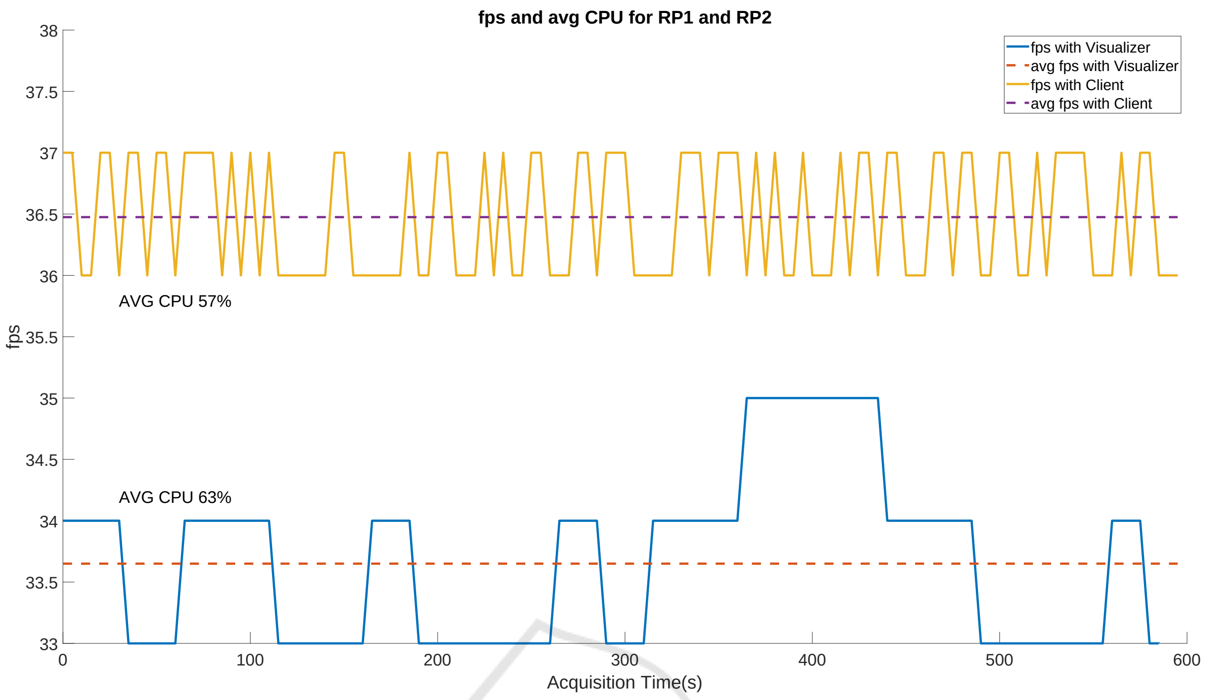


Figure 6: RP1 performance, in terms of frame rate and average CPU usage. Both parameters were sampled every 5 seconds for a total of 600 seconds.

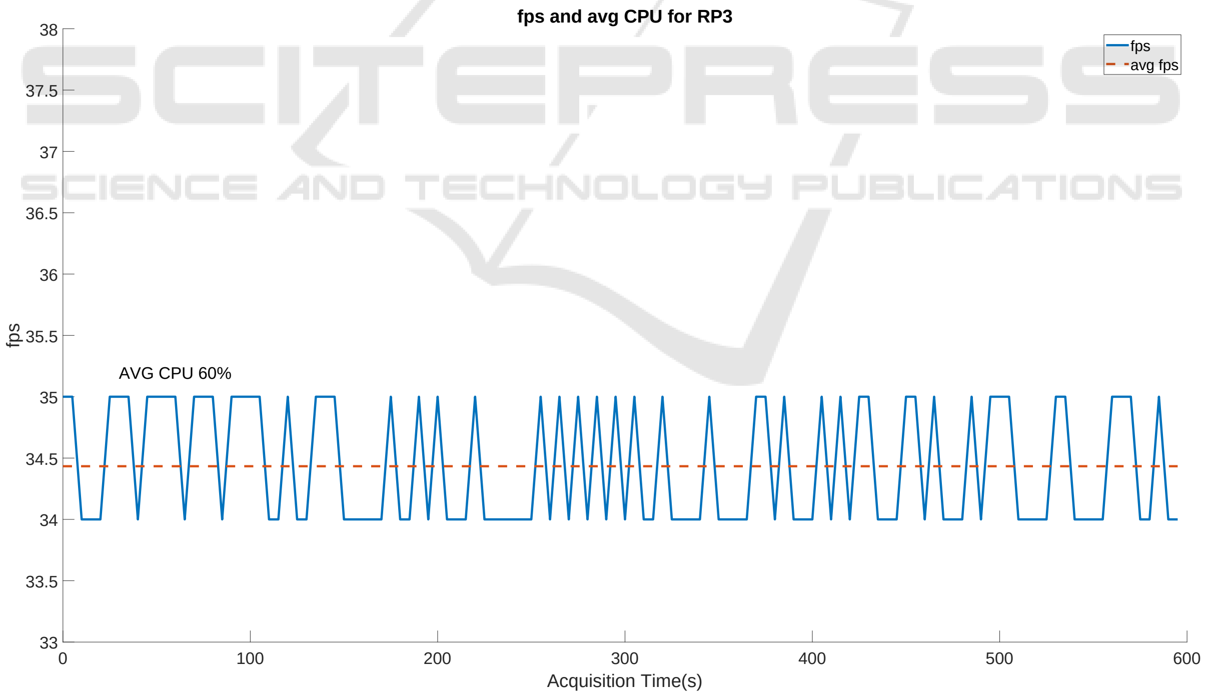


Figure 7: RP3 performance, in terms of frame rate and average CPU usage. Both parameters were sampled every 5 second for a total of 600 seconds.

der about 34 fps in the web-based rendering software with an average CPU usage of 60%. The final frame rate is lower than that in RP1 and RP2 because the merging operation among the models coming from RP1 and RP2 is time consuming, though RP3 was free from the driver of the LEAP device. Also in this case, the CPU is not completely used and further optimization is possible through, for example, the use of smart shape-based segmentation strategies (Franchi et al., 2009). however, for our purposes, the goal of maintaining a frame rate of at least 30 fps is completely fit.

It is important to note that a fundamental role is assumed by the used SD: in fact, when we tried to change the SD with one of the same capacity but of 80 MB/s in transfer rate, the performance fell to about 17 fps. This fact authorized us to imagine that the use an SD with a transfer rate greater than 120 MB/s could contribute to improve the frame rate above 35 fps, though this has not been attempted and it is out the scope of our project (our goal is to obtain a final frame rate which is above 30fps in the final hand model reproduced in a virtual environment).

However, we have verified that the hardest role in our project is assumed by RP3: in fact, its assigned tasks are data fusion, model reconstruction and reproduction in the graphic interface. The final frame also using the graphic interface was 34 fps when using the smart fusion strategy presented in (Placidi et al., 2021). Though still acceptable, that number is very close to the lower limit, though a further gain in fps can be obtained by stressing the CPU usage.

The final assembly, showing RPi-Box, LEAP support, wide-screen, keyboard and mouse is reported in Figure 4.

In this final version, keyboard and mouse were used: a touch-screen could allow the elimination of both these devices.

5 CONCLUSION

We have demonstrated that a VG can be implemented by using cheap PCs instead of a costly, bulky and power consuming PC. In fact, we have implemented a low-cost and compact embodiment of the VG by using three light RP 4 model B. The results is a VG version with a frame rate of 34 fps which is acceptable for most of VG purposes, though it could be too low for high (temporal) resolution procedures (such as medical interventions). Improvements could be obtained along different directions:

1. by using last, most powerful, versions of RP;
2. by using faster SDs;
3. by optimizing the software in order to better use the CPU power;
4. by using the 64 bit version of the emulator when it will be available;
5. by using more than three RP.

The last case could improve performance by dividing the tasks of RP3 among two RPi in series: one to collect data from RP1 and RP2 and to fuse the models and the other to render the final model in the VR environment.

Thanks to the scalability of the system, a further extension of the proposed architecture would be to drive more than two LEAP sensors, maybe by implementing the designs proposed in (Kiselev et al., 2019; Shen et al., 2019), for further reducing occlusions. In that case, however, the RP acting as a hub would receive and process information from several RPi and an efficient and smart data fusion strategy would be necessary to maintain real-time.

REFERENCES

Ameur, S., Ben Khalifa, A., and Bouhlel, M. S. (2020). A novel hybrid bidirectional unidirectional lstm network for dynamic hand gesture recognition with leap motion. *Entertainment Computing*, 35:1–10.

Ankit, C., Jagdish, R. L., Karen, D., and Sonia, R. (2011). Intelligent approaches to interact with machines using hand gesture recognition in natural way: A survey. *International Journal of Computer Science & Engineering Survey*, pages 122–133.

Avola, D., Spezialetti, M., and Placidi, G. (2013). Design of an efficient framework for fast prototyping of customized humancomputer interfaces and virtual environments for rehabilitation. *Computer Methods and Programs in Biomedicine*, 110(3):490–502.

Bachmann, D., Weichert, F., and Rinkenauer, G. (2014). Evaluation of the leap motion controller as a new contact-free pointing device. *Sensors*, 15(1):214–233.

Battaglia, E., Bianchi, M., Altobelli, A., Grioli, G., Catalano, M. G., Serio, A., Santello, M., and Bicchi, A. (2015). Thimblesense: a fingertip-wearable tactile sensor for grasp analysis. *IEEE transactions on haptics*, 9(1):121–133.

Carrieri, M., Petracca, A., Lancia, S., Moro, S. B., Brigadoi, S., Spezialetti, M., Ferrari, M., Placidi, G., and Quaresima, V. (2016). Prefrontal cortex activation upon a demanding virtual hand-controlled task: A new frontier for neuroergonomics. *Frontiers in Human Neuroscience*, 10(53):1–13.

Chen, S., Ma, H., Yang, C., and Fu, M. (2015). Hand gesture based robot control system using leap motion. In *International Conference on Intelligent Robotics and Applications*, pages 581–591. Springer.

Choi, P. J., Oskouian, R. J., and Tubbs, R. S. (2018). Telesurgery: Past, present, and future. *Cureus*.

Erden, F. and Cetin, A. E. (2014). Hand gesture based remote control system using infrared sensors and a camera. *IEEE Transactions on Consumer Electronics*, 60(4):675–680.

Franchi, D., Gallo, P., Marsili, L., and Placidi, G. (2009). A shape-based segmentation algorithm for x-ray digital subtraction angiography images. *Computer Methods and Programs in Biomedicine*, 94(3):267–278.

Imran, J. and Raman, B. (2020). Deep motion templates and extreme learning machine for sign language recognition. *The Visual Computer*, 36(6):1233–1246.

Jin, H., Chen, Q., Chen, Z., Hu, Y., and Zhang, J. (2016). Multi-leapmotion sensor based demonstration for robotic refine tabletop object manipulation task. *CAAI Transactions on Intelligence Technology*, 1.

Kiselev, V., Khlamov, M., and Chuvilin, K. (2019). Hand gesture recognition with multiple leap motion devices. In *2019 24th Conference of Open Innovations Association (FRUCT)*, pages 163–169. IEEE.

Liu, Y. and Zhang, Y. (2014). Toward welding robot with human knowledge: A remotely-controlled approach. *IEEE Transactions on Automation Science and Engineering*, 12(2):769–774.

Luzhnica, G., Simon, J., Lex, E., and Pammer, V. (2016). A sliding window approach to natural hand gesture recognition using a custom data glove. In *2016 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 81–90. IEEE.

Mahdikhanlou, K. and Ebrahimnezhad, H. (2020). Multimodal 3d american sign language recognition for static alphabet and numbers using hand joints and shape coding. *Multimedia Tools and Applications*, 79(31):22235–22259.

Marin, G., Dominio, F., and Zanuttigh, P. (2016). Hand gesture recognition with jointly calibrated leap motion and depth sensor. *Multimedia Tools and Applications*, 75(22):14991–15015.

Moro, S. B., Carrieri, M., Avola, D., Brigadoi, S., Lancia, S., Petracca, A., Spezialetti, M., Ferrari, M., Placidi, G., and Quaresima, V. (2016). A novel semi-immersive virtual reality visuo-motor task activates ventrolateral prefrontal cortex: a functional near-infrared spectroscopy study. *Journal of Neural Engineering*, 13(3):1–14.

Mueller, F., Deuerlein, C., and Koch, M. (2019). Intuitive welding robot programming via motion capture and augmented reality. *IFAC-PapersOnLine*, 52(10):294–299.

Oudah, M., Al-Naji, A., and Chahl, J. (2020). Hand gesture recognition based on computer vision: A review of techniques. *Journal of Imaging*, 6(8):73.

Petracca, A., Carrieri, M., Avola, D., Basso Moro, S., Brigadoi, S., Lancia, S., Spezialetti, M., Ferrari, M., Quaresima, V., and Placidi, G. (2015). A virtual ball task driven by forearm movements for neuro-rehabilitation. In *2015 International Conference on Virtual Rehabilitation (ICVR)*, pages 162–163.

Placidi, G. (2007). A smart virtual glove for the hand telerehabilitation. *Computers in Biology and Medicine*, 37(8):1100 – 1107.

Placidi, G., Avola, D., Cinque, L., Polsinelli, M., Theodoridou, E., and Tavares, J. M. R. S. (2021). Data integration by two-sensors in a LEAP-based virtual glove for human-system interaction. *Multimedia Tools and Applications*, 80(12):18263–18277.

Placidi, G., Avola, D., Iacoviello, D., and Cinque, L. (2013). Overall design and implementation of the virtual glove. *Computers in Biology and Medicine*, 43(11):1927–1940.

Placidi, G., Cinque, L., Petracca, A., Polsinelli, M., and Spezialetti, M. (2017). A virtual glove system for the hand rehabilitation based on two orthogonal leap motion controllers. In *Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM,*, pages 184–192. INSTICC, SciTePress.

Placidi, G., Cinque, L., Polsinelli, M., and Spezialetti, M. (2018). Measurements by a leap-based virtual glove for the hand rehabilitation. *Sensors*, 18(3):1–13.

Placidi, G., Petracca, A., Spezialetti, M., and Iacoviello, D. (2015). A modular framework for EEG web based binary brain computer interfaces to recover communication abilities in impaired people. *Journal of Medical Systems*, 40(1).

Shen, H., Yang, X., Hu, H., Mou, Q., and Lou, Y. (2019). Hand trajectory extraction of human assembly based on multi-leap motions. In *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 193–198.

Smith, R., Cucco, E., and Fairbairn, C. (2020). Robotic development for the nuclear environment: Challenges and strategy. *Robotics*, 9(4):94.

Voulodimos, A., Doulamis, N., Doulamis, A., and Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, 2018:1–13.

Wang, Z., Wang, D., Zhang, Y., Liu, J., Wen, L., Xu, W., and Zhang, Y. (2020). A three-fingered force feedback glove using fiber-reinforced soft bending actuators. *IEEE Transactions on Industrial Electronics*, 67(9):7681–7690.

Yang, L., Chen, J., and Zhu, W. (2020). Dynamic hand gesture recognition based on a leap motion controller and two-layer bidirectional recurrent neural network. *Sensors*, 20:2106–2123.

Zhang, W., Cheng, H., Zhao, L., Hao, L., Tao, M., and Xiang, C. (2019). A gesture-based teleoperation system for compliant robot motion. *Applied Sciences*, 9(24):1–18.