Reducing dizziness when using a video-see-through head-mounted display

Segovia Barreales, Richard

Resum– Resum del projecte, màxim 10 línies. ....

Paraules clau– Paraules clau del treball, màxim 2 línies . ...

Abstract– Versió en anglès del resum . ...

Keywords– Accomodation vergence conflict, video-see-through, head mounted display, depth map, stereo calibration.

1 Introduction

The development of the technologies related with head mounted displays (HMD) has grown in the recent years mainly centered in the video-games field, some examples are the Oculus [7] or the HTC Vive [3]. These devices are called virtual reality headsets because they are only capable of showing computer generated scenes.

Despite that the industry is mainly focused developing virtual reality systems, there are two kinds of HMD that allow the visualization of the environment surrounding the user.

- The optical-see-through use image projectors that display the image over a see-thought mirror, hence allowing the user to see computer generated images over the environment. An example of this kind of devices is the Microsoft HoloLens [11].
- The video-see-through devices use one or two cameras placed in the front of the headset and show the stream of images in two screens places in front of the eyes. This project uses these kind of devices, one of the prototypes can be seen in Fig.1.

• E-mail: richard.segovia@e-campus.uab.cat
• Menció en Computació
• Project supervised by: Coen Antens (CVC) and Felipe Lumbreras (Computació)
• Course 2017/18

Fig. 1: Current version of the HMD video-see-through prototype.

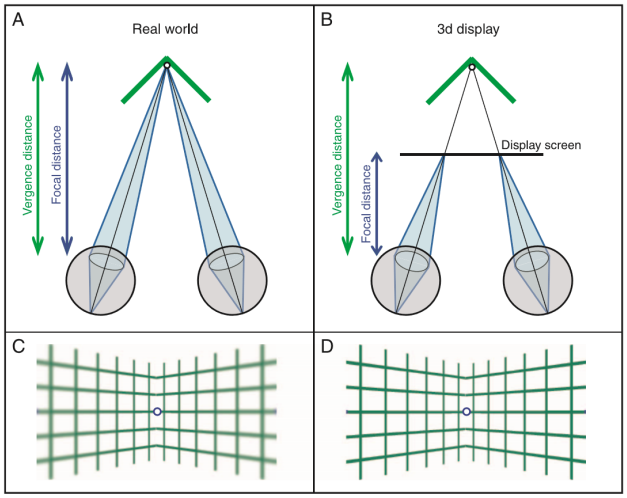

Fig. 2: In A and C can be seen the usual effect that happens when the coupling of the accommodation vergence is correct. As the accomodation distance is the same as the vergence distance, a blur effect appears in the corners. In B and D can be seen the effect that happens when an stereo scene is showed through near eye screens. As the accomodation distance is different than the vergence distance, no blur effect is applied in the corners. Source [6].

Both types of devices can show stereo images, this allow the user to sense depth in the displayed images.

This project originated from the need to establish and resolved the reasons why the users have a poor experience, dizziness and eye strain, when using video-see-through devices. As is explained in [14] and was experimentally tested in [6], the Accommodation-Vergence conflict is one of the issues that causes a poor user experience.

The vergence is the process where the eyes set their angle of visualization trying to fuse the image of an object keeping it into sharp focus, whereas the accommodation is the process where the objects difficult to fuse are blurred. Both process are tightly coupled giving each other feedback in order to keep the images as sharp and fused as possible.

The issue arises as a result of using near eye screens to show stereo 3D. In these displays the image is shown always at the same distance of the eye, however, the distance between objects, disparity, changes when the environment is changed. This conflict can be seen in Fig.2

Consequently, this project tries to solve the Accommodation-Vergence conflict using the environment changes information to set the distance between images of the viewer.

2 State of the art

The Accomodation-Vergence conflict is a topic of great interest in the research field, therefore, a wide variety of solutions have been proposed to try to solve this problem.

One of the many solutions found to ease this problem is applying blur to zones where the image should not fuse, thus simulating the effect that the image would have had if the conflict would not have happened, see [10]. Related with this, a research [12] found that placing objects that connect different depth planes helps the users to better transition between objects in different planes and aids to maintain the coupling between accommodation and vergence. Other line of research is to use eye-tracking techniques [4], these technologies allow the system to know where the eye is pointing over the image and then blur the out of focus areas.

After all of this, we can see that this project takes a different approach from the main lines of research. This project uses depth data collected from a stereo camera placed in front of the headset to change the distance between images inside the headset. However, to get the depth information a reliable and fast stereo matcher was needed. That is why ELAS [1] [5] and its implementation LIBELAS were used.

In the stereo matcher field, two main branches can be found, Local stereo matchers and global stereo matchers. Global stereo matchers are reliable but use much compute power. On the contrary, local stereo matchers are fast but less reliable than the global matchers. ELAS uses first a global stereo matcher on highly reliable points and after removing the more redundant, a Delaunay triangle mesh is created from that points. After that, the regions are then processed by a local stereo matcher. More information about ELAS implementation can be found on [5].

3 Objectives

After the previous analysis and explanation of the problem, the main objectives will be the following:

- Evaluate the user experience when using the HMD and determine whether the accommodation-vergence effect causes dizziness and general discomfort on the users.
- If the Accomodation-Vergence is one of the causes of the discomfort on the users, the problem will be solved using the deph information that can be obtained using stereo vision.
- Evaluate the user experience after the development and conclude if our approach for solving this problem has reduce the discomfort on the users.
- Add the required modules keeping in mind the usefulness of these for future developments.

[1] Efficient large-scale stereo matcher

- Make all the required changes in the visualization program without reducing the performance.

4 METHODOLOGY

Scrum and its variants are one of the most spread work methodologies nowadays. Therefore we believe convenient to use it as one of the foundation of this project, however, as this project was only done by one person, some changes were made.

As there was only one developer, the daily meeting was replaced with a weekly meeting with the project supervisors, in this case the tutor and the boss of the laboratory department in the CVC. In these meetings we evaluated the development done, the issues faced that week and the problems solved. Related with the scrum methodology, Github [8],[9] was used as version control system and trello[1] as the task manager system.

Involving the tools used to develop this project, C++ and QT [2] were used mainly because the previous development of the visualization system was done using that environment. QT libraries were used to develop the interface and the visualization of the images on the screen. In addition to that, OpenCV was used as the library for image processing and calibration, [13]. Some code snippets for testing and evaluation of the results were done using Matlab[15].

5 TOOLS AND DEVELOPMENT

First the tools and modules used will be presented and after the pipeline and integration of the parts will be explained.

5.1 Calibration

The calibration module is key inside the pipeline to be able to get the LIBELAS depth map. This module can be split in several parts:

First, a dataset of captures of a chess like pattern have to be taken to get references of the distortion and distance of the cameras. In these images the pattern has to be visible by both cameras and cannot be occluded neither be partially out of the image. In addition to that, the dataset must contain images of the pattern in various positions, the more locations covered by the images better will be the results of the calibration.

Second, once the dataset is taken, the next step is locating the chess pattern inside each image, to locate these points a OpenCV function is used first to find them and after to make these points more precise. In this step each camera dataset will be processed independently.

Once the points are calculated, these are passed to the monocular calibration function of OpenCV to obtain the intrinsic matrix and the distortion vector. Although these parameters per se can be used in a rectification function to undistort the images, we need a stereo rectification that also rectifies the traslation and the rotation between cameras leaving the epipolar lines aligned. For that reason, a second phase of calibration is needed.

After the monocular calibration, these matrices along with the image points are passed to the stereo calibration function and the resulting matrices passed again to the rectification function. Once this function is finished we now have the projection and rotation matrices from both cameras, these can be passed to the undistortion function along with two images to rectify the distortion and align both images in the same epipolar lines, alternatively, these matrices can be passed to a undistort mapping function for a faster undistortion of the images.

Some problems were faced during the development of this module. The main was that some adjustment had to be done in the configuration parameters of the OpenCV function because excessive undistortion was done over the images leaving them completely unusable. The parameters adjusted were the number of coefficients from the distortion vector used by the calibration modules.

All of these matrices and configurations can be saved and loaded to avoid the need to calibrate each time the program is started.

5.2 Libelas

This module starts with two images that have already been undistorted with the stereo calibration, these then are passed to the LIBELAS processing function, this function computes the disparity from two given images and returns two disparity maps, one for each given image. These maps indicate us the distance between one pixel from one image to the other.

One of the problems faced by this library, and in general by depth map stereo matchers, is the existence of low textured areas on the image, as will be seen in **??** these regions usually end up without disparity values, therefore, potentially producing issues in further processing. These regions along with occluded ones and calibration faulty areas are usually set to negative values by Libelas itself, therefore can be easily detected.

These maps can also be visualized in grayscale by changing the range of the image or to a colormap to better appreciate the depth gradient.

One issue faced during the development of this module was the extremely high calibration precision needed in order to get quality images. The cameras, not totally locked in position, usually moved slightly, reducing the quality of the depth map obtained. The issue was communicated to the hardware development team in order to change the HDM prototype to better secure the cameras.

Although LIBELAS is a fast library it was not quick enough to process the images in real time or near real time, to solve this issue, two possible solutions were found.

- The first idea was to apply a ROI in the center of the images after the distortion, that way as the image is smaller, it requires less computing power to be processed. In addition to that, it will keep the center of the image with full detail, where usually the user will be looking at.

- Other idea was to make down sampling of the images before using them on LIBELAS. This also decreases the computing power required to process the images, however it also decreases the detail of the image, potentially affecting the quality of the depth map.

In **??** comparison between both methods can be seen.

To improve the quality of the depth map obtained in LIBELAS the default configuration called "Robotics" was taken as base and modified. First, we added the configuration that enables support points to be in the corners. We also removed the postprocessing features like median filtering or mean filtering. These last changes improve the speed without affecting the performance of the classification, as will be seen in **??** we already use mean operations to calculate the classification values.

5.3 Configuration

As the users have vision systems physically distinct between them, different eye sizes, different distance between eyes; each user has to have their own customized configuration to better fit their needs. To be able to use these, two modules were developed.

First, a module able to modify the position of the images over the screens of the HMD was developed. This module is designed to show only a ROI of the full resolution image, letting the user decide and change every possible aspect of this ROI, from the size of the ROI to the anchor point. These parameters can also be saved to be used in other sessions or to be able to change between different settings set for different distances. Once the ROI is obtained graphic functions from QT are called to display them on the screen.

Second, a module capable of make transitions between settings was developed. This module calculates the distance between two settings and does the transition in a given number of steps.

5.4 Pipeline and module integration

Parallel to the development of said modules, a pipeline was built to be able to achieve the main goal of the project.

A simple classifier has been develop to determine at which distance the user is currently viewing. It uses the output of LIBELAS and performs an operation (mean, max or min) after that, the value is used on a threshold classifier to select between 3 different distances, near, medium and far. To assure stability on the output of the classifier the output of the last classifications is preserved and a mean of these last values along with the current is done and used as the classification output.

As can be seen in Fig.3 the pipeline can be split in the following parts:

- Grabbing threads: these threads grab images from the cameras, that are set to be continuously grabbing. After that, the images are then stored on a variable overwriting the previous image. This thread is continuously looping to grab the latest image.

- Processing thread: this threat is set to use an image set by the displaying thread, this image will be first undistorted and then passed to LIBELAS to obtain the depth map. After that the central region is crop and the image is passed to the classificator. Then, the output classified distance is stored on a variable which will be queried by the displaying thread on each iteration.

- Displaying threads: the goal of this thread is to get the images from the grabbing threads and display them on the screen of the HMD. For this it uses the ROI module in combination with the setting system. This module also passes the images grabbed to the image processing thread, and uses its output, the classified distance, to dynamically select between user settings. Also, if set, it will use a transition to change between settings.

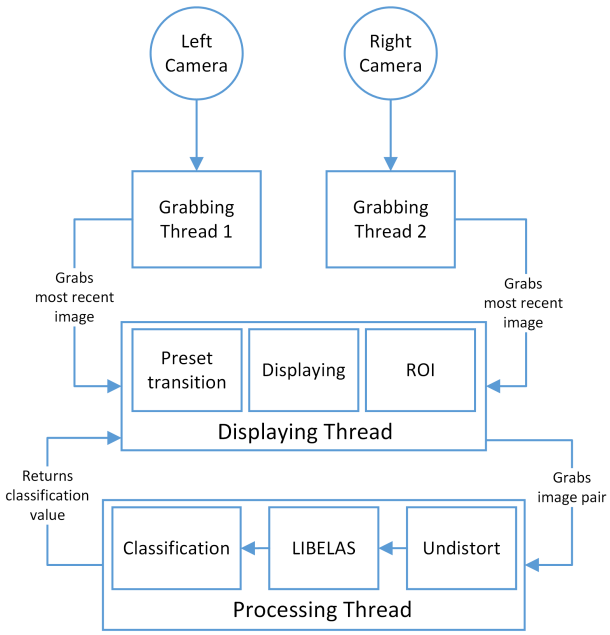

Fig. 3: Diagram of the pipeline of the system. At top the grabbing threads that capture the images from the cameras. In the middle the thread that displays the images and changes between user settings. At the bottom the thread that process the images and classifies them.

It was decided to use a threaded architecture to allow to each thread to perform their tasks at their own pace without harming the performance of the displaying thread. This helps to reduce the reaction time improving the user experience.

5.5 Offline pipeline

One of the issues had was that libelas was not able to produce real time processing, as is explained in 5.4. For that reason, a video recording module was also developed.

The module was developed using opencv and the videos were intended to be saved in raw format, but due to an unresolved bug in the opencv modules that prevented the reading of raw videos, the videos were saved instead using the Intel's IYUV lossless codec.

After the implementation of this module an offline pipeline was developed to allow the processing of these videos. The architecture is similar to the online pipeline seen in **??** thought has some noteworthy differences.

First, the images are read from a video instead of a camera, as the frames from the video can be grab on demand, every frame can be processed. The input video can be already undistorted if that is not the case, the images are undistorted using the calibration module explained in **??**

Second, after the frame is processed by LIBELAS, the output depth map can be converted to a grayscale or colormap image and be saved in another video for further pro-

cessing. Otherwise the images can be evaluated in place using the threshold classificator already explained.

6 RESULTS

6.1 Libelas

6.1.1 ROI vs resize

6.1.2 resize time performance

6.2 Classsification

6.3 performance stability

6.4 First user testing

aqui expondremos los resultados de la primera prueba de user testing que se hizo, explicaremos nuestras conclusiones previas sobre los resultados de la prueba y nuestras propuestas para tratar de mejorar los resultados

6.5 Second user testing

en este apartado mostraremos los distintos resultados obtenidos en la segunda sesion de user testing , explicaremos los resultados y concluiremos si nuestra hipotesis es correcta y si la solucion desarrollada es suficiente para resolver este problema, en caso de que no, nos plantearemos cuales son o han sido los problemas que impiden que el usuario sienta una mejora al utilizar la vergencia dinamica.

7 CONCLUSIONS

Finalmente expondremos todo el trabajo realizado y apartir de los resultados de las sesiones de user testing explicaremos si se han logrado los objetivos y cuanto margen de mejora hay en caso de haberlo.

8 FUTURE WORK

(quizas esto es mas para la presentacion, nose si en el informe tambien se deberia de poner) en este apartado explicaremos ideas que se plantearon y que no llegaron a realizarse y ideas con las que podria continuarse este proyecto, (DoF blur, third camera, Augmented reality)

ACKNOWLEDGMENT

This work is supported in part by a CVC transfer project with ProCare Light company, and partially funded by the Spanish Ministry of Economy and Competitiveness and FEDER under grants TIN2014-56919-C3-2-R and TIN2017-89723-P.

REFERENCES

[1] Atlassian. Trello. https://trello.com, 2018. Last access March 9th 2018.

[2] The QT Company. Qt. https://www.qt.io/, 2018. Last access March 9th 2018.

[3] HTC Corporation. Vive. https://www.vive.com/us/, 2018. Last access March 10th 2018.

[4] Andrew T Duchowski, Donald H House, Jordan Gestring, Rui I Wang, Krzysztof Krejtz, Izabela Krejtz, Radosław Mantiuk, and Bartosz Bazyluk. Reducing visual discomfort of 3d stereoscopic displays with gaze-contingent depth-of-field. In *Proceedings of the acm symposium on applied perception*, pages 39–46. ACM, 2014.

[5] Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient large-scale stereo matching. In *Asian Conference on Computer Vision (ACCV)*, 2010.

[6] David M Hoffman, Ahna R Girshick, Kurt Akeley, and Martin S Banks. Vergence–accommodation conflicts hinder visual performance and cause visual fatigue. *Journal of vision*, 8(3):33–33, 2008.

[7] Facebook Inc. Oculus rift. https://www.oculus.com/, 2018. Last access March 10th 2018.

[8] GitHub Inc. Github. https://github.com/, 2018. Last access March 9th 2018.

[9] GitHub Inc. Github desktop. https://desktop.github.com/, 2018. Last access March 9th 2018.

[10] Robert Konrad, Nitish Padmanaban, Keenan Molner, Emily A Cooper, and Gordon Wetzstein. Accommodation-invariant computational near-eye displays. *ACM Transactions on Graphics (TOG)*, 36(4):88, 2017.

[11] Microsoft. Microsoft hololens. https://www.microsoft.com/es-es/hololens, 2018. Last access June 19th 2018.

[12] Steven Poulakos, Gerhard Roethlin, Adrian Schwaninger, Aljoscha Smolic, and Markus Gross. Alternating attention in continuous stereoscopic depth. In *Proceedings of the ACM Symposium on Applied Perception*, pages 59–66. ACM, 2014.

[13] Opencv team. Opencv library. https://opencv.org/, 2018. Last access March 10th 2018.

[14] Kasim Terzic and Miles Hansard. Causes of discomfort in stereoscopic content: a review. *arXiv preprint arXiv:1703.04574*, 2017.

[15] Inc. The MathWorks. Matlab. https://es.mathworks.com/products/matlab.html, 2018. Last access March 22th June 2018.

A OBJECTIVE AND TASKS LIST

lista de tareas y objetivos (similar a lo que tenia en las otras entregas)

B ADDITIONAL IMAGES

En este apartado se incluiran imagenes extra de ejemplo(mas escenarios) i/o imagenes que no quepan en el documento en si

C USER TESTING PROTOCOL