POLITECNICO DI TORINO

Master in Computer Engineering

Master Thesis

Oculus Rift and AR

Supervisors:
prof. Antonio Lioy

Candidato:
Giovanni Pautasso

Dicembre 2015

Sommario

Inserire qui un breve sommario della tesi.

Ringraziamenti

Opzionali, solo nel caso si sia ricevuto un aiuto speciale e particolarmente rilevante.

Indice

Capitolo 1

Introduction

1.1 Taxonomy

A VR/AR setup must deal with different disciplines and integrate them fluently. We will now analyse the most important aspects the work described in the next chapter takes from, belonging to the following fields of expertise:

- computer vision, which covers the techniques for displaying/enhancing images and extrapolating numerical or symbolic information about the real world. Optics and imaging sensors in general are modelled and therefore used to extract that data;
- stereoscopy, a collection of techniques addressing the specific problem of creating or enhancing the illusion of depth in images, exploiting principles in optics and human perception. Most methods involve two separate images to achieve this effect. A combination of such techniques and computer vision models opens the door to computer stereo vision, where more than one image is used to extract data, thus requiring less known parameters from the scene;
- virtual reality, a discipline that implies mathematical models, actuators and sensors to replicate real world experiences in form of entirely computer-simulated environments and give them interaction capabilities. High relevance is given to human psychology and perception limits in order to recreate a lifelike experience. Images are in this case entirely synthetic: models and image enhancements implied have lot in common with computer vision, although the latter focuses on analysis. We find the most common expression of their combination in what is called augmented reality.

1.2 Augmented reality and Augmented Virtuality

1.2.1 Optical and Video See-through devices

1.3 Immersivity

1.3.1 Resolution

1.3.2 Frame Rate

1.3.3 Refresh Rate

1.3.4 Latency

Capitolo 2

Related work and state of the art

2.1 Camera modelling fundamentals in computer vision

2.1.1 Pinhole camera

2.1.1.1 intrinsic parameters

2.1.2 Lens distortion correction and camera calibration

2.1.2.1 distortion parameters

2.1.2.2 extrinsic parameters

2.1.3 Fish-eye camera

2.2 Solutions for stereoscopy

2.2.1 Stereopsis

2.2.2 Stereo rig configurations

2.2.2.1 toed-in cameras

2.2.2.2 parallel cameras

2.2.2.3 shifted-lens cameras

2.2.3 Stereo camera calibration

2.2.4 From standard to fish-eye stereo

2.3 Modelling first person view in VR

2.3.1 Virtual camera: projection and view matrix principles

2.3.2 Head model: IPD and ETN parameters

2.3.3 Rendering pipeline for HMDs

2.3.4 Tackling lag: Oculus Rift timewarping

Capitolo 3

Proposed solution

3.1 Gear selection

3.1.1 HMD and head tracking

3.1.2 Camera for stereo

low cost best resolution/fps rate

3.1.3 3D Engine for VR environment

OculusSDK Ogre3D

3.1.4 Libraries for CV and AR

OpenCV + tbb + cuda arUco

3.2 Camera image pipeline

3.2.1 Frame timing and pose estimation

TIME
Approximate: simply get time before grab new frame
Precise-auto: compute image-to-grab delay from frame timestamp (if available)
Precise-manual: set image-to-grab delay manually
POSE
Approximate: get the current pose before grab new frame
Predicted: predict the pose in the past by a delay
Precise: get the current pose a delay before calling grab()

3.2.2 Image elaboration

3.2.2.1 Lens distortion correction

3.2.2.2 Marker detection

reference mapping between Ogre and arUco
pivot offset to place the object on the marker
undistort tolerance

3.2.2.3 Image effect/enhancement

3.2.2.4 A Cuda-ready pipeline

3.2.3 Frame synchronization

3.3 3D Rendering pipeline

3.3.1 Enhanced head model

3.3.2 Virtual eye render mapping to real eye

3.3.3 Camera image mapping to virtual eye

Suppose camera is in the same position of the eye (as aruco)
aruco detected object must be placed relative to the scene (after detected)

better marker detection with two cameras? who cares

3.3.3.1 pin-hole camera model

3.3.3.2 fish-eye camera model

3.3.4 Image stabilization

Image pose application
Head stabilization
Eye stabilization

3.4 Stereo Rig design

alignment of cameras is critical!

3.4.1 wide-angle lens configuration

TOED-IN (-) edge violation, see through a window effect (lower immersion) (-) keystoning effect at edges, but reducible (+) compensate the low stereo cues area of parallel config (+) easy to implement (+) good vertical disparities (VSR)

3.4.2 fish-eye lens configuration

PARALLEL with high fov (+) no edge violation (close objects are welcome, high immersion) (+) no keystoning (+) no need to toe-in to compensate low fov.. (-) .. but still can be very distorted at edge (need high quality lens!) (-) no stereo cues at the edge of the image (-) hard to implement (due to additional undistortion) (+) good vertical disparities (VSR)

3.4.3 3D printed prototype (?)

Capitolo 4

Experimental results

Capitolo 5

Conclusions and future work

5.1 Increasing immersivity

5.1.1 Using a Skybox

5.1.2 Employing Fish-eye lenses

Increased percieved FOV
Decouple from camera FPS

5.1.3 Virtual nose

5.2 Performance optimizations for future applications

5.2.1 Low FPS: decreasing image resolution

5.2.2 Undistort and marker detection in CUDA

5.2.3 Mono mode

5.3 Further development

5.3.1 Depth mapping with stereo computer vision

5.3.2 Introducing virtual hands with Leap Motion

5.3.3 Integration with ROS