

Gaze Estimation with Pupil Tracking in Real-Time

Tomas Bartipan
Supervisor: Benjamin Busam

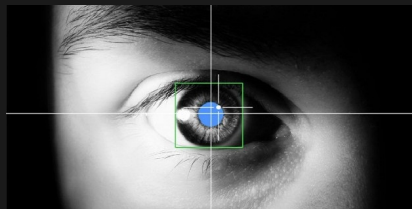
Technical University of Munich
Chair of Computer Aided Medical Procedures
Project Management and Software Development for Medical Applications

November 9, 2017

tomas.bartipan@tum.de

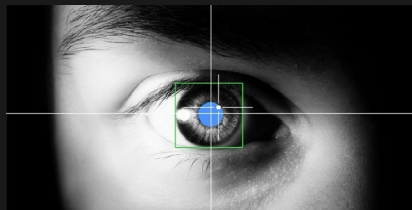
Motivation for Gaze Estimation

- Diagnosis of the eye
- Psychology
- Marketing and design
- Automotive
- Visual-input modality



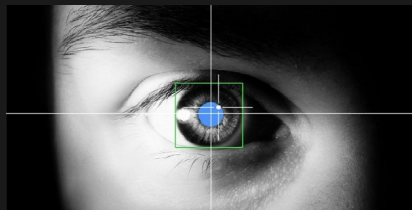
Motivation for Gaze Estimation

- Diagnosis of the eye
- **Psychology**
- Marketing and design
- Automotive
- Visual-input modality



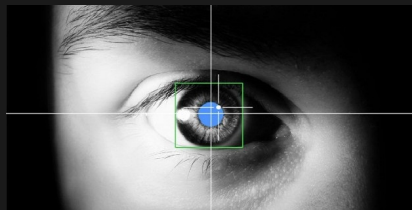
Motivation for Gaze Estimation

- Diagnosis of the eye
- Psychology
- Marketing and design
- Automotive
- Visual-input modality



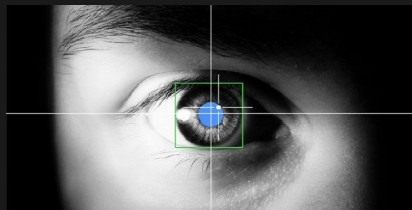
Motivation for Gaze Estimation

- Diagnosis of the eye
- Psychology
- Marketing and design
- **Automotive**
- Visual-input modality



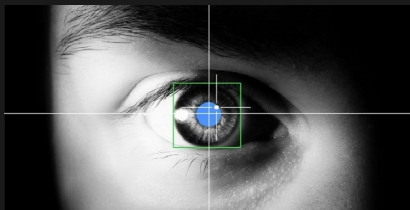
Motivation for Gaze Estimation

- Diagnosis of the eye
- Psychology
- Marketing and design
- Automotive
- **Visual-input modality**



Motivation for Gaze Estimation

- Diagnosis of the eye
- Psychology
- Marketing and design
- Automotive
- Visual-input modality
 - Especially for AR



The method, hardware and software

Method

Corneal reflections - find the rotation of the eye based on reflection of Near-Infrared (NIR) light using irregularities in its spherical shape

Hardware

Smartek NIR/Grayscale 1600x1200px, 25fps camera

Falcon ring NIR LED illumination

LAN switch and a computer running Windows

Software

FRAMOS C++ framework

MATLAB for intrinsic camera calibration

The method

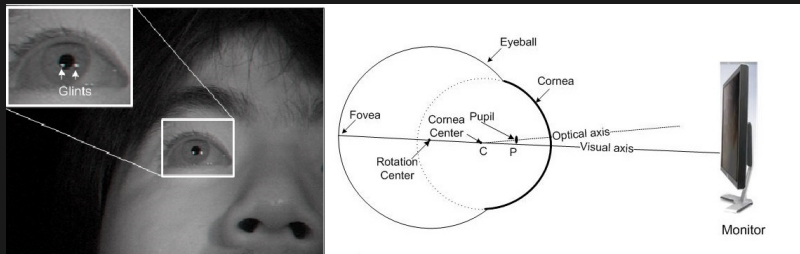


Figure: NIR light reflection and eye schematic

Source:

Real-Time Eye Gaze Tracking Under Natural Head Movements
<https://www.ecse.rpi.edu/~cvrl/zhiwei/gazetracking/gazetracking.html>

Accessed: 9/11/17

The hardware



Figure: Hardware in use

The software

The screenshot displays the FRAMOS toolkit interface. The main window is titled "Framos GmbH Mirror Tracking Synthetic". It features a menu bar (File, Configuration, Serialization, Runmode, Language, Processing, Window, View, Help) and a toolbar with icons for file operations and execution. The interface is divided into several panels:

- 3dDisplay:** A 3D visualization window showing a black background with a green and blue wireframe sphere.
- getPts:** A task graph editor showing a flowchart with nodes for "If Condition", "Init Variable", "Sequence Control", "Logging", and multiple "Read Data" tasks.
- Status:** A table showing the status of various tasks.
- Logging:** A table showing log entries with Date and Time.
- Property:** A panel for configuring task properties.
- Task info:** A panel for task information.
- Configur:** A panel for configuration.
- Blackboard:** A table listing variables and their types.
- Interfaceboard:** A table listing interface components.
- Sensorboard:** A table listing sensor components.

The bottom status bar shows "User name: service User level: SERVICE" and a timestamp of "0, 0 11.9.2017 13:00:40".

Status	Task	Date	Time
0	dRMSMirror	2017.11.09	12:59:25.155
0	dRMSUS	2017.11.09	12:59:24.957
0	dRMSUSviaMirror	2017.11.09	12:59:15.814
0	nCountPtsSeq	2017.11.09	12:59:15.026
0		2017.11.09	12:59:15.026
0		2017.11.09	12:58:06.782
0		2017.11.09	12:58:06.782
0		2017.11.09	12:58:06.699
0		2017.11.09	12:58:05.798
0		2017.11.09	12:58:05.798
0		2017.11.09	12:58:05.688

Name	Type
Mirror	mesh
MirrorDisplay	mesh
US	mesh
readInput	mesh
startCom	mesh
stopCom	mesh
bCom	bool
test	f
task	bool
do?	bool
writeData	bool
bFitRejected	bool
bUSviaMirrorVisible	bool

Name	Vendor	Model
igtLinkClientMirror		
igtLinkClientUS		
igtLinkClientUSvia...		
testClient		

Figure: FRAMOS toolkit

The task at hand

- 1 Get to know the FRAMOS C++ framework
- 2 Research possible solutions considering hardware limitations
- 3 Pick one of the existing solutions
- 4 Implement iris-detection
- 5 Implement gaze-estimation
- 6 Present a prototype at PMSD and FRAMOS

The task at hand

- 1 Get to know the FRAMOS C++ framework
- 2 **Research possible solutions considering hardware limitations**
- 3 Pick one of the existing solutions
- 4 Implement iris-detection
- 5 Implement gaze-estimation
- 6 Present a prototype at PMSD and FRAMOS

The task at hand

- 1 Get to know the FRAMOS C++ framework
- 2 Research possible solutions considering hardware limitations
- 3 Pick one of the existing solutions**
- 4 Implement iris-detection
- 5 Implement gaze-estimation
- 6 Present a prototype at PMSD and FRAMOS

The task at hand

- ① Get to know the FRAMOS C++ framework
- ② Research possible solutions considering hardware limitations
- ③ Pick one of the existing solutions
- ④ **Implement iris-detection**
- ⑤ Implement gaze-estimation
- ⑥ Present a prototype at PMSD and FRAMOS

The task at hand

- ① Get to know the FRAMOS C++ framework
- ② Research possible solutions considering hardware limitations
- ③ Pick one of the existing solutions
- ④ Implement iris-detection
- ⑤ **Implement gaze-estimation**
- ⑥ Present a prototype at PMSD and FRAMOS

The task at hand

- 1 Get to know the FRAMOS C++ framework
- 2 Research possible solutions considering hardware limitations
- 3 Pick one of the existing solutions
- 4 Implement iris-detection
- 5 Implement gaze-estimation
- 6 (Optional) Realize the solution isn't that great, start again
- 7 Present a prototype at PMSD and FRAMOS

The task at hand

- 1 Get to know the FRAMOS C++ framework
- 2 Research possible solutions considering hardware limitations
- 3 Pick one of the existing solutions
- 4 Implement iris-detection
- 5 Implement gaze-estimation
- 6 (Optional) Realize the solution isn't that great, start again
- 7 Present a prototype at PMSD and FRAMOS

Optimistic timeline

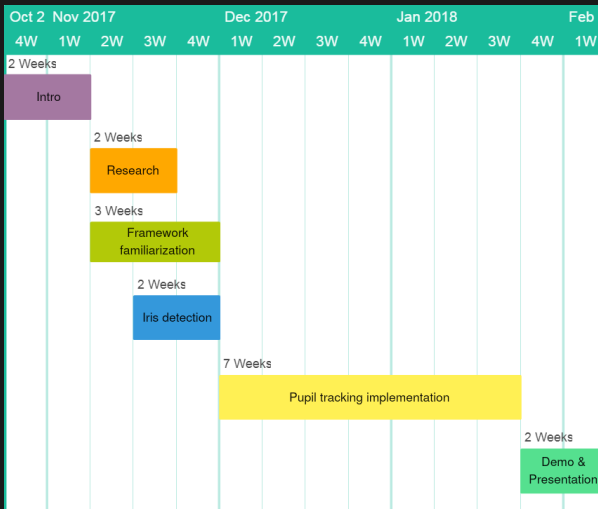


Figure: Gantt time-chart

That's it, folks!

Questions?

Eye picture @ slide 2 credit :

<https://www.geek.com/news/>

[eye-tracking-mario-demo-could-offer-a-glimpse-at-the-next-generation-of-google-glass-1577302/](https://www.geek.com/news/eye-tracking-mario-demo-could-offer-a-glimpse-at-the-next-generation-of-google-glass-1577302/)

Accessed 9/11/17