PA198
Augmented Reality Interfaces

Lecture 7
Augmented Reality Registration and Calibration

Fotis Liarokapis
liarokap@fi.muni.cz

06th November 2017

Image and Video Registration

Image and Video Registration

- Image and video registration is a deeply studied area from Computer Vision
- It is concerned with the alignment of image sequences with respect to each other
  - i.e. in a common 3D coordinate system
- Many applications ranging from ubiquitous panorama creations to sophisticated interfaces for interaction with media

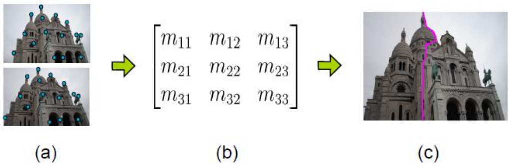
http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Image Registration Common Pipeline

- Suppose we have a sequence of images we want to put in the same coordinate system
  - First, we need to identify features in the images and find correspondences between pairs of images
  - Then, we can estimate the transformation relating the image planes
  - This estimation must be robust to outliers due to mismatched features
  - Finally, we optimize the position of the images taking the whole set into account, for better results

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Image Registration Common Pipeline .


m_{11} m_{12} m_{13}
m_{21} m_{22} m_{23}
m_{31} m_{32} m_{33}
(a)
(b)
(c)

(a) Feature detection and matching
(b) Transformation estimation between two images
(c) Image stitching with seam optimization (illustrative only)

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Detection and Matching

- Consider two frames of a video and the task of creating a panorama from them
- While there is much coherence in the two images, the information about how they would fit together is not explicit
- The similarity of a part of the first image with a part of the second image needs to be formalized in a way that a computer can understand it
- This lecture will focus on the detection and matching of two types of features:
  - Regions
  - Lines

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Points and Patches

- Points and patches are the most common features used when trying to find correspondences between images
  - They provide the easiest algorithms
- Two approaches can be used in point matching:
  - Finding features that can be tracked in a set of images
    - Recommended when the scene does not change much
  - Finds the most prominent features of each image and then tries to match the sets

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Points and Patches .

- Four main steps:
  - Feature detection
  - Feature description
  - Feature matching
  - Feature tracking

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

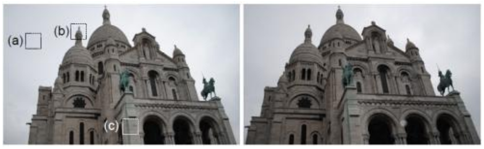
Feature Detection

- Detect which parts of the image make the best features
  - To avoid ambiguities
- The question that comes up when confronted with the task of selecting a set of features from an image is what makes a good feature?
- The most distinct patches should be selected

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Detection .

- Different patches of the image
  - (b) good feature
  - (a), (c) bad features



http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Detection ..

- The simplest way to match two features form different images is to take the sum of squared differences between them
- Can be described as:

$$Error(u) = \sum [I_1(x_i + u) - I_0(x_i)]^2$$

- where
  - *I_0* and *I_1* are the images being compared
  - *u* is a displacement vector

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Detection ...

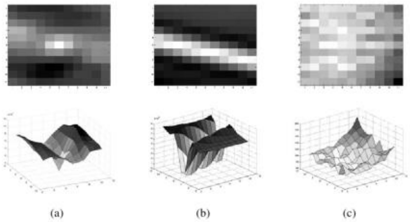
- Next, evaluate how different a patch is from the patches around it using an autocorrelation function

$$AutoCorrelation(u) = \sum [I_0(x_i + u) - I_0(x_i)]^2$$

- Note that the displacement vector u here would represent how the second patch is displaced from the original one

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Detection ....



Three types of features and the respective autocorrelation values (error is plotted in terms of the displacement) (a) a good feature with a distinct minimum (b) a feature suffering from aperture (c) a very bad feature for which many points in the neighbourhood are minima

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Detection .....

- We can approximate the function by using the first terms of a Taylor Series expansion to generate an autocorrelation matrix that will then be used to define how distinctable a feature is
- After simplifications:

$$\begin{aligned} AutoCorrelation(u) &= \sum [I_0(x_i + u) - I_0(x_i)]^2 \\ &\approx \sum [I_0(x_i) + \nabla I_0(x_i)u - I_0(x_i)]^2 \\ &= \sum [\nabla I_0(x_i)u]^2 \\ &= \sum u^T A u \end{aligned}$$

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Detection ......

- The matrix A is called the autocorrelation matrix and its properties can be used to decide whether a feature is distinct or not

$$A = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix}$$

- How the properties of the autocorrelation matrix are used to determine whether a feature should be used or not depends on the type of detection we do

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Detection ....

- Some examples:
  - Shi and Tomasi
  - Förstner-Harris
  - Adaptive non-maximal suppression
  - SIFT

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Shi-Tomasi Example 1

https://www.youtube.com/watch?v=jKW2y5NhO_k

Shi-Tomasi Example 2

https://www.youtube.com/watch?v=IFAQBgC8F08

Feature Description

- Once we have decided which parts of the image would generate good features, we need to match them
- A simple approach would be minimizing the sum of squared differences
- Unfortunately, this would only work if the transformation applied to the patch was a simple translation
- In most cases, the patch will undergo arbitrary affine transformations that will require more sophisticated methods for comparison
  - See next slide

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Description .

- MOPS (Multi-Scale Oriented Patches)
  - Used in applications that do not require invariance for transformations
  - It consists of sampling a lower frequency version of the image around the feature
  - Patch intensities are re-scaled such that their mean is zero and the variance is 1
- GLOH (Gradient Location-Orientation Histogram)
  - Extension of SIFT that uses polar bins instead of square ones
  - Divides the space into 3 radial bins and 6 angular bins - with one additional bin for radius = 0

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Description ..

- SIFT (Scale-invariant feature transform)
  - Descriptors are formed by computing the gradient in a 16x16 window around the pixel
  - The contribution of each gradient is weighted by a Gaussian centered at the pixel
  - The 16x16 window is divided into sixteen 4x4 regions, and represents the gradient in these regions by adding them to 8 bins
  - The 128 resulting values (the values in each of the 8 bins, in each of the 16 regions, 16x8 = 128) are the SIFT descriptor

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Matching

- There are two important steps when matching features
  - Define a matching strategy
  - Combine data structures and algorithms to efficiently perform the matching evaluations

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Matching .

- A simple approach is to immediately reject matches that are further away from each other than a threshold value (using Euclidean distance)
  - In this case, we must of course observe that the threshold is consistent with our expected camera motion, to avoid false positives and false negatives
  - The problem with this strategy is that the threshold depends on each case and is difficult to optimize

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Matching ..

- A different approach is to use nearest neighbors in feature space
  - Since some features would not have matches, a threshold approach is still used to avoid extreme false positives

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Tracking

- An alternative approach to detecting candidate features in each image and then matching them is to find features in the first image and track them throughout the image sequence
- This approach is most commonly used in videos
  - Because the scene is not expected to change much from one frame to the next
  - The definition of a good feature to track is the same as before
- If features are being tracked in long image sequences, it is important to consider the problem of big changes in the patches, as well as the loss of patches by occlusion

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

Feature Tracking .

- To solve this problem, features are matched in different images along the sequence using an affine motion model
  - Instead of a simple translational one
- Since affine matching is more expensive than translational, a first estimation is performed using simpler models
- The resulting tracker is called Kanade-Lucas-Tomasi, or KLT

http://www.inf.ufrgs.br/~rgschneider/research/reports/videoRegistration.pdf

KLT AR Example

https://www.youtube.com/watch?v=8VbylCRn3iI

AR Registration

AR Registration

- Registration is the accurate positioning of virtual information into the real environment
- Mis-registration:
  - Breaks the illusion that the two coexist
  - Prevents acceptance of many serious applications

Video Key Method

- Video Keying is a process that is widely used in television, film production and CG. (weather report)
- When using video keying to design AR scenes, one signal contains the foreground image and the other one contains the background image
- The 'keyer' combines the two signal to produce a combined video which is then sent to the display device

Video Key Method .

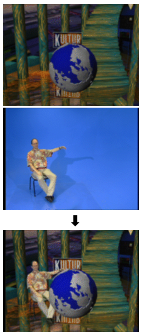
- Keying can be done using composite or component video signals
  - A composite video signal contains information about color, luminance, and synchronization, thus combining three piece of information into one signal
  - With component video, luminance synchronization are combined, but chroma information is delivered separately

Video Key Method ..

- Chroma keying involves specifying a desired foreground key color
- Foreground areas containing the keying color are then electronically replaced with the background image
- This results in the background image being replaced with the fore ground image in areas where the background image contains chroma color
- Blue is typically used for chroma keying (Chromakey blue) rarely shows up in human skin tones

Video Key Method ...

- If a video image of the real world is chosen as the foreground image, parts of the scene that should show the computer-generated world are rendered blue
- In contrast, if video of the real world is chosen as the background image, the computer generated environment will be located in the foreground



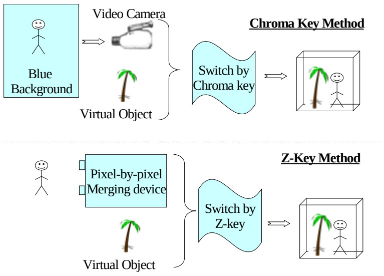
Chroma Keying AR Example

https://www.youtube.com/watch?v=FOeVKSC2XA8

Z-key Method

- The z-key method requires images with both depth information (depth map) as inputs
- The z-key switch compares depth information of two images for each pixel, and connects output to the image which is the nearer one to the camera
- The result of this is that real and virtual objects can occlude each other correctly
- This kind of merging is impossible by the chroma-key method, even if it is accompanied with some other positioning devices such as magnetic or acoustic sensor, since these devices provide only a gross measurement of position

Chroma Key vs Z-Key



Chroma Key Method

Blue Background

Video Camera

Virtual Object

Switch by Chroma key

Z-Key Method

Pixel-by-pixel Merging device

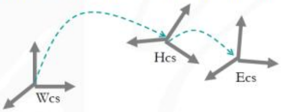
Virtual Object

Switch by Z-key

MASARYKOVA UNIVERZITA

Spatial Registration

- Defining Relative Position of Each Elements of a Scene



- Elements are: User, User's eye, Environment (Table, Room, Building), Objects, etc.
- Coordinate Systems (Euclidian System)
- Initially: Calibration
- Temporally: 3D/6D **Tracking**

Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

MASARYKOVA UNIVERZITA

Sources of Registration Errors

- Static errors
  - Optical distortions
  - Mechanical misalignments
  - Tracker errors
  - Incorrect viewing parameters
- Dynamic errors
  - System delays (largest source of error)
    - 1 ms delay = 1/3 mm registration error

Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

MASARYKOVA UNIVERZITA

Reducing Static Errors

- Distortion compensation
- Manual adjustments
- View-based or direct measurements
- Camera calibration

Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

MASARYKOVA UNIVERZITA

View Based Calibration (Azuma 94)

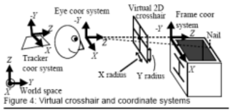


Figure 4: Virtual crosshair and coordinate systems

Figure 1: Wooden frame for calibration and registration

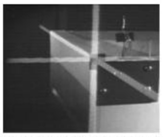
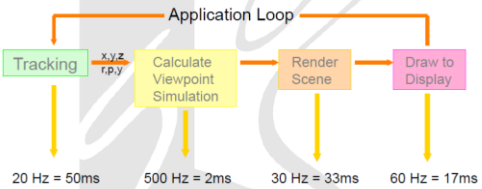


Figure 2: View seen in HMD, virtual axes on real frame

Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

MASARYKOVA UNIVERZITA

Dynamic Errors



Application Loop

Tracking → x,y,z r,p,y → Calculate Viewpoint Simulation → Render Scene → Draw to Display

20 Hz = 50ms

500 Hz = 2ms

30 Hz = 33ms

60 Hz = 17ms

- Total Delay = 50 + 2 + 33 + 17 = 102 ms
  - 1 ms delay = 1/3 mm = 33mm error

Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

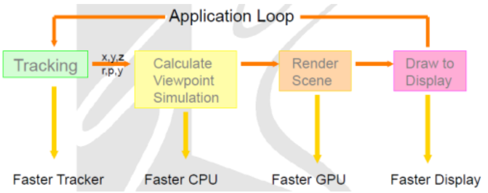
MASARYKOVA UNIVERZITA

Reducing Dynamic Errors

- Reduce system lag
  - Faster components/system modules
- Reduce apparent lag
  - Image deflection
  - Image warping

Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

Reducing System Lag


Application Loop

Tracking

x,y,z
r,p,y

Calculate Viewpoint Simulation

Render Scene

Draw to Display

Faster Tracker

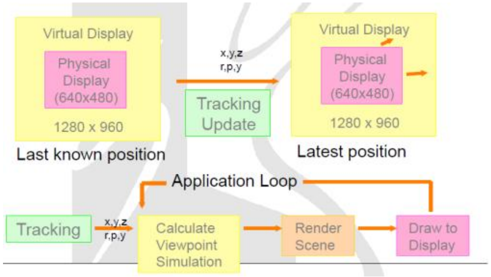
Faster CPU

Faster GPU

Faster Display

Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

Reducing Apparent Lag


Virtual Display

Physical Display (640x480)

1280 x 960

Last known position

x,y,z
r,p,y

Tracking Update

Virtual Display

Physical Display (640x480)

1280 x 960

Latest position

Application Loop

Tracking

x,y,z
r,p,y

Calculate Viewpoint Simulation

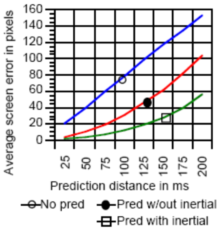
Render Scene

Draw to Display

Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

More on Reducing Dynamic Errors

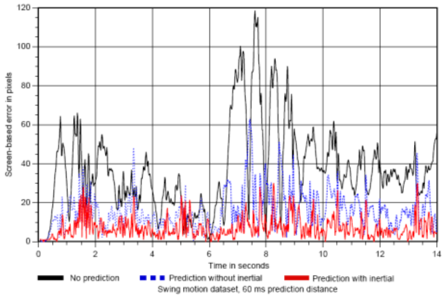
- Match input streams
  - Delay video of real world to match system lag
- Predictive Tracking
  - Inertial sensors helpful



Azuma / Bishop 1994

Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

Predictive Tracking (Azuma 94)



Billinghurst, M. COSC 426: Augmented Reality, July 26th 2013.

AR Calibration

Calibration Requirements

- "Ideally, the calibration methods should be statistically robust, there should be a variety of approaches for different circumstances, and metrology equipment should be sufficiently accurate, convenient to use, and not too expensive"
  - (Hollerbach and Wampler, 1996)

Additional Calibration Requirements

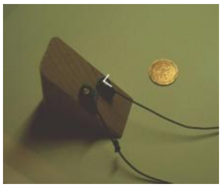
- Additional Requirements of Experimental Subsystems:
  - Independent
    - Not rely on each other
  - Subject-specific
    - Account for individual differences
  - Avoid residual cues
    - To prevent subjects using them in unanticipated ways

Typical Calibration Components

- Workspace
- Point of view
- Physical objects

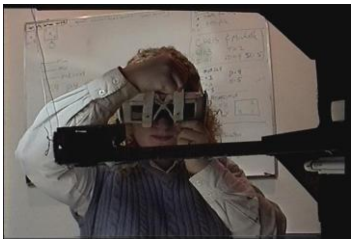
Workspace Calibration

- Markers are aligned with virtual crosses
- Exactly one position in 3-space eliminates “swim”
- do NOT need stereo to calibrate



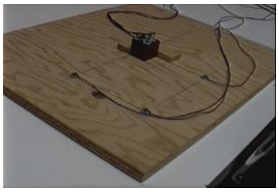
Point of View Calibration

- Subject placing eye calibration bars



Physical Object Calibration

- Markers placed anywhere on object
- Place object in frame so XYZ orientations match



Measuring ARToolKit’s Tracking Error

- In wide area applications, the positioning accuracy of ARToolKit is not very robust
- In distances between 1m and 2.5m the error in the x and y values increases proportionally with the distance from the marker
- Calculate error in distances ranging between 20 cm and 80 cm under normal lighting conditions

Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

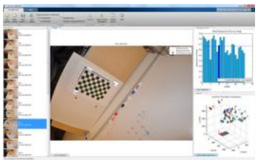
Camera Calibration with MATLAB

- Computer Vision System Toolbox™ provides an app and functions to perform all essential tasks in the camera calibration workflow, including:
  - Fully automatic detection and location of checkerboard calibration pattern including corner detection with subpixel accuracy
  - Estimation of all intrinsic and extrinsic parameters including axis skew
  - Calculation of radial and tangential lens distortion coefficients
  - Correction of optical distortion
  - Support for single camera and stereo calibration

https://www.mathworks.com/videos/camera-calibration-with-matlab-81233.html?requestedDomain=www.mathworks.com

Camera Calibrator App

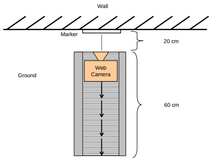
- Used to select and filter calibration images, choose the number and type of radial distortion coefficients, view reprojection errors, visualize extrinsic parameters, and export camera calibration parameters



https://www.mathworks.com/videos/camera-calibration-with-matlab-81233.html?requestedDomain=www.mathworks.com

Measuring ARToolKit's Tracking Error .

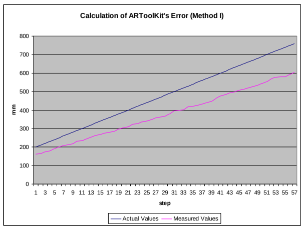
- The optimal area, which contains the least error, is the one that is perpendicular to the marker card
- A rigid path is set so that the camera can not loose its direction while moving backwards
- Numerous measurements of the location of the web camera in a local co-ordinate system



Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

Measuring ARToolKit's Tracking Error ..

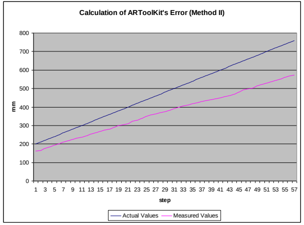
- Error is proportional to the distance



Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

Measuring ARToolKit's Tracking Error …

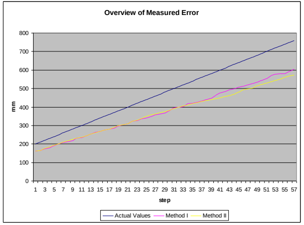
- Camera facing the marker at variable angle (yaw) having the other two (pitch, roll) stable



Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

Measuring ARToolKit's Tracking Error ….

- Differences in the error produced from the experiments compared with the actual values



Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

Calculating Camera Parameters

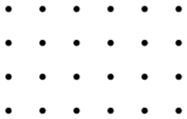
ARToolKit's Calibration Method

- ARToolKit provides two software tools called calib_dist and calib_param that can be used to calculate these camera properties
  - calib_dist is used to measure the lens distortion and the image centre point
  - calib_param is used to compute the focal length of the camera

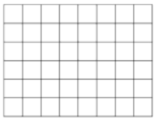
Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

ARToolKit's Calibration Method .

- In calib_dist program, an image of a pattern 6x4 dots spaced equally apart is used



Camera calibration for lens distortion

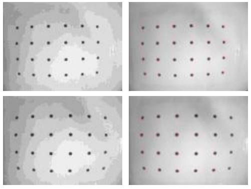


Camera calibration for focal length

Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

KLT Tracking Method

- A feature-based vision method is known as the KLT (Kanade-Lucas-Tomasi) algorithm based on a model or affine image changes



Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

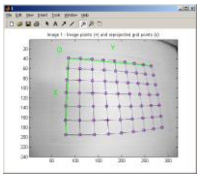
Camera Calibration Toolbox for Matlab

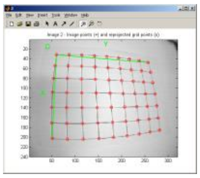
- Offers an automatic mechanism for counting the number of squares in each grid
- All calibration images are searched and focal and distortion factor are automatically estimated

	Estimated focal (pixels)	Estimated distortion factor (kc)
Image 1	*201.2503*	*-0.2*
Image 2	*224.1415*	*-0.2*
Image 3	*231.7859*	*-0.2*
Image 4	*114.2254*	*-0.055*
Image 5	*265.8596*	*-0.28*
Image 6	*211.5925*	*-0.2*

Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

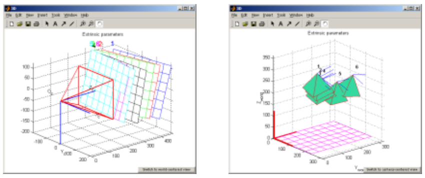
Calibration Without Lens Distortion


File Edit View Insert Tools Window Help
Image 1 - Image points (+) and reprojected grid points (o)
O
Y
X
20
40
60
80
100
120
140
160
180
200
220
240
50
100
150
200
250
300


File Edit View Insert Tools Window Help
Image 2 - Image points (+) and reprojected grid points (o)
O
Y
X
20
40
60
80
100
120
140
160
180
200
220
240
50
100
150
200
250
300

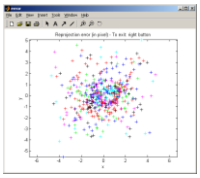
Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

Calculation of Extrinsic Parameters

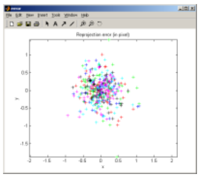


Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

Error Calculation of Extrinsic Camera Parameters



Initial Error



Corrected Error

Liarokapis, F., Augmented Reality Interfaces - Architectures for Visualising and Interacting with Virtual Information, Sussex theses S 5931, Department of Informatics, School of Science and Technology, University of Sussex, Falmer, UK, 2005

Outdoor Markerless Tracking and Registration in AR

https://www.youtube.com/watch?v=hRRAGuL8lVk

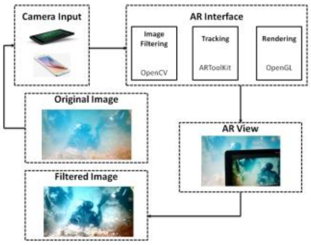
Underwater Dehazing

Underwater Vision

- Vision underwater is degraded by several factors, mostly by turbidity and absorption of lighting, which is different for different color channels
- Concerning the turbidity in the underwater images, the problem is very similar to dehazing
  - i.e. to correct the effects of fog and haze in images taken above water

Bruno, F., Lagudi, A., Ritacco, G., Agrafiotis, P., Skarlatos, D., Čejka, J., Kouřil, P., Liarokapis, F., Philpin-Briscoe, O., Poullis, C., Mudur, S., Simon, B. Development and integration of digital technologies addressed to raise awareness and access to European underwater cultural heritage. An overview of the H2020 i-MARECULTURE project, OCEANS 2017, IEEE Computer Society, Aberdeen, Scotland, 19-22 June 2017.

AR Dehazing Architecture



Bruno, F., Lagudi, A., Ritacco, G., Agrafiotis, P., Skarlatos, D., Čejka, J., Kouřil, P., Liarokapis, F., Philpin-Briscoe, O., Poullis, C., Mudur, S., Simon, B. Development and integration of digital technologies addressed to raise awareness and access to European underwater cultural heritage. An overview of the H2020 i-MARECULTURE project, OCEANS 2017, IEEE Computer Society, Aberdeen, Scotland, 19-22 June 2017.

Implementation

- In the first step, the input image from tablet camera is filtered and improved to reduce defects and other imperfections caused by turbidity or poor lighting conditions
- This pre-processed image serves in the second step as an input for detection of objects in real world for augmented reality
- In the third step, objects of augmented reality are composed together with improved pre-processed input image and rendered into output image
- This image is then displayed on the screen of the tablet

Bruno, F., Lagudi, A., Ritacco, G., Agrafiotis, P., Skarlatos, D., Čejka, J., Kouřil, P., Liarokapis, F., Philpin-Briscoe, O., Poullis, C., Mudur, S., Simon, B. Development and integration of digital technologies addressed to raise awareness and access to European underwater cultural heritage. An overview of the H2020 i-MARECULTURE project, OCEANS 2017, IEEE Computer Society, Aberdeen, Scotland, 19-22 June 2017.

Implementation .

- This prototype was implemented in Java and is focused for Android platforms
- The input from the camera feed is in NV21 format. NV21 is a variant of the YUV420 format
- The image is kept in this format, and in the pre-processing step, Contrast Limited Adaptive Histogram Equalization (CLAHE) (Pizer et al, 1987) is applied to its Y channel
- This process leaves the U and V channels unchanged
- The implementation of the CLAHE algorithm was based on the OpenCV library (OpenCV)
- The augmented reality part is handled by ARToolKit library (ARToolKit)

Bruno, F., Lagudi, A., Ritacco, G., Agrafiotis, P., Skarlatos, D., Čejka, J., Kouřil, P., Liarokapis, F., Philpin-Briscoe, O., Poullis, C., Mudur, S., Simon, B. Development and integration of digital technologies addressed to raise awareness and access to European underwater cultural heritage. An overview of the H2020 i-MARECULTURE project, OCEANS 2017, IEEE Computer Society, Aberdeen, Scotland, 19-22 June 2017.

Results

Bruno, F., Lagudi, A., Ritacco, G., Agrafiotis, P., Skarlatos, D., Čejka, J., Kouřil, P., Liarokapis, F., Philpin-Briscoe, O., Poullis, C., Mudur, S., Simon, B. Development and integration of digital technologies addressed to raise awareness and access to European underwater cultural heritage. An overview of the H2020 i-MARECULTURE project, OCEANS 2017, IEEE Computer Society, Aberdeen, Scotland, 19-22 June 2017.

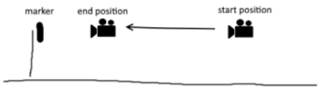
Results .

- The AR application was tested on several videos taken in underwater environment
- The videos were played on a monitor of a PC, and a tablet was used to record, process and display the images
- The resolution of input camera was 1280 x 720
- We performed our tests on two devices, NVIDIA Shield K1 tablet and Samsung Galaxy S6 phone
- The pre-processing step took 7.8 milliseconds on the tablet and 9.0 milliseconds on the phone, which in both cases allowed the application to present improved images in real-time (Bruno et al., 2017)

Bruno, F., Lagudi, A., Ritacco, G., Agrafiotis, P., Skarlatos, D., Čejka, J., Kouřil, P., Liarokapis, F., Philpin-Briscoe, O., Poullis, C., Mudur, S., Simon, B. Development and integration of digital technologies addressed to raise awareness and access to European underwater cultural heritage. An overview of the H2020 i-MARECULTURE project, OCEANS 2017, IEEE Computer Society, Aberdeen, Scotland, 19-22 June 2017.

Dehazing for Marker-based Tracking

- Testing scenarios for evaluation of impact of methods for dehazing images to marker-based tracking



D6.3, iMARECULTURE Underwater AR Platform, Public Document, Oct 2017

Underwater Tracking

Data Collection

- Obtained a set of videos from underwater sites in Cyprus at 10m depth

Video 1

Video 2

Video 3

Video 4

Video 5

Video 6

Implementation

- For improving images, we used CLAHE, [87CLAHE], which is a simple and fast method that runs real-time, and a method based on a fusion of results of several algorithms described in [12Enhancing] (further referenced as Fusion)
- Implementation of another method based on bright channel prior described in [16Restoration] (further referenced as BCP) is in development

Implementation .

- For marker detection, we used three real-time open source libraries
  - ARToolKit [ARToolKit] is one of the best known open source library, which can be used to detect markers with generic patterns
  - ARUco [ARUco] is a library based on OpenCV and its functions, and is used to robustly detect markers with binary matrix codes
  - AprilTags [AprilTags] is a library developed at the University of Michigan, and is focused on robust detection of markers in different lighting conditions and angles of view

Results

Video 1	CLAHE	Fusion	Detection time (ms)
ARToolKit	24 / 0 / 3	17 / 0 / 10	2365
ARUco	446 / 22 / 543	58 / 51 / 931	8074
AprilTags	535 / 25 / 442	72 / 10 / 905	62686
Video 2	**CLAHE**	**Fusion**	**Detection time (ms)**
ARToolKit	0 / 0 / 0	0 / 0 / 0	3800
ARUco	6 / 3 / 59	0 / 7 / 65	10637
AprilTags	11 / 40 / 10	0 / 39 / 21	87523
Video 3	**CLAHE**	**Fusion**	**Detection time (ms)**
ARToolKit	0 / 0 / 0	0 / 2 / 0	3272
ARUco	47 / 5 / 71	3 / 2 / 115	9533
AprilTags	5 / 60 / 20	1 / 65 / 24	88634
Video 4	**CLAHE**	**Fusion**	**Detection time (ms)**
ARToolKit	39 / 3 / 386	9 / 23 / 416	1175
ARUco	11 / 21 / 425	7 / 43 / 429	3280
AprilTags	4 / 3 / 424	0 / 1 / 428	91260
Video 5	**CLAHE**	**Fusion**	**Detection time (ms)**
ARToolKit	6 / 150 / 50	19 / 47 / 37	1308
ARUco	35 / 104 / 323	131 / 7 / 227	5714
AprilTags	84 / 66 / 271	133 / 67 / 222	136616
Video 6	**CLAHE**	**Fusion**	**Detection time (ms)**
ARToolKit	0 / 5 / 2	0 / 3 / 2	1321
ARUco	2 / 0 / 2	0 / 0 / 4	3022
AprilTags	0 / 1 / 0	0 / 0 / 0	25539

Comparison of marker detection algorithms (ARToolKit, ARUco, AprilTags) and image enhancing methods (CLAHE, Fusion)

The result of each measurement are three numbers

The first number is the number of markers detected in the original video, but not in the enhanced video

The second number is the number of markers detected in the enhanced video, but not in the original video

The third number is the number of markers detected in both videos

In the rightmost column, we can see the time needed to detect markers in the whole video

Results .

- The results show us that in some videos tracking is improved by enhancing the image, while in other videos, tracking is worsen by enhancing the image
- This differs with different marker detectors and different image enhancement methods
- However, in most videos, most non-damaged markers were found in the original and processed images, significant differences were in damaged and occluded markers
- For this reason, such markers must be discarded

Results ..

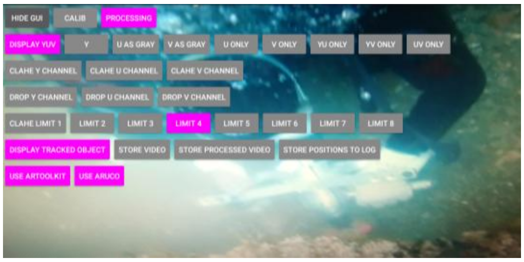
- We can also see that processing times of individual marker detectors
- Videos were processed on a desktop computer with Intel Core i5 760 processor and 8 GB operating memory
- We can clearly see that ARToolKit finds the markers in the lowest amount of time, while AprilTags needs the most time
- We assume the results to be comparable when running on mobile devices, although the processing times presented in the table is measured on a desktop computer

Underwater Registration

AR Interface

- We designed the first interface for an application that recorded input video in original quality without performing any lossy compression, performed a simple image enhancing, calibrated the camera, and detected markers

1st Prototype



Implementation

- This design allowed setting of various parameters from the user interface
  - i.e. It allowed to set which color channel is enhanced, the parameters of image enhancing algorithm, turning marker detection on and off etc
- The interface also included a button which hid the interface when everything was set, and shown it again when parameters needed to be changed

Implementation .

- All parameters are saved into a text file onto the internal disk and reloaded when the application is started next time
- Several parameters (for example a path where the recorded videos are saved) cannot be set from the application though; they can be set only in this text file
- This allowed us to simplify the GUI

Testing

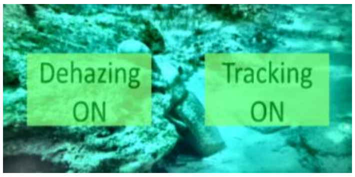
- This user interface was tested in April 2017 by CUT in Cyprus
- This testing showed that it is very hard to manipulate the application with this interface, due to very small buttons
- Also, we found that most of the parameters of this application were set once and did not need to be changed when the application run.

2nd Prototype

- Results from the tests of the first application helped us with the design of the next applications
- The main changes are:
  - We create many specialized applications rather than one application containing every feature to reduce the complexity of the user interface.
  - We use only a very few buttons places through the whole screen to ease the manipulation with the interface in underwater conditions.
  - We place parameters of the application into a text file, leaving only a few parameters to be changed directly in the application

Interface Example

- This application improves the image presented to the user, tracks markers, and displays virtual objects



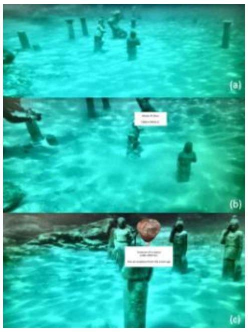
Implementation

- The user interface contains only two buttons, which allows the user to turn on/off dehazing (an algorithm for enhancing the input image), and to turn on/off tracking of markers
- The buttons span the whole left and right part of the screen, though their visual representation is smaller
- This makes the manipulation with the interface much easier
- The user interface also hides itself automatically after a few seconds of user's inactivity, reappearing again when the user touches the screen
- All other parameters (parameters of the image improving algorithm, parameters of tracking and others) can be set only from a text file that contains application parameters

Implementation .

- The application is capable of rendering 3D objects and text descriptions at places of these markers
- Currently it demonstrates the capabilities of the application
- In future, it will be used to augment the vision of the user by adding virtual aids into the scene, show artefacts similar to those seen on the screen, or show the appearance of the objects in the past
- It will also be used to provide the user with a description of seen objects or places

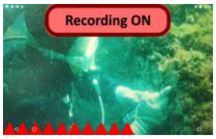
Results



AR presentation underwater
(a) the underwater environment
(b) textual augmentation
(c) textual description and 3D information

Results .

- A second application for recording video and storing its raw data without using and lossy compression is created
  - This application does not do any image enhancements and does not track markers



Results ..

- This interface contains two indicators
  - One indicator is place in the top part of the screen, and represents the state of the recording
  - The other indicator is in the bottom part of the screen, and informs the user about the speed of recording
- Additionally, the whole screen behaves as a single button, controlling the state of recording
- The interface is always visible; it cannot be hidden in any way
  - Similarly to the previous interface, all other parameters are set from a text file

Questions