Erasmus Mundus Joint Master Degree in Image Processing and Computer Vision (EMJMD-IPCV)

Applied Video Sequence Analysis

Lab 4 “Histogram-based object tracking”

Collaborative work (part II)

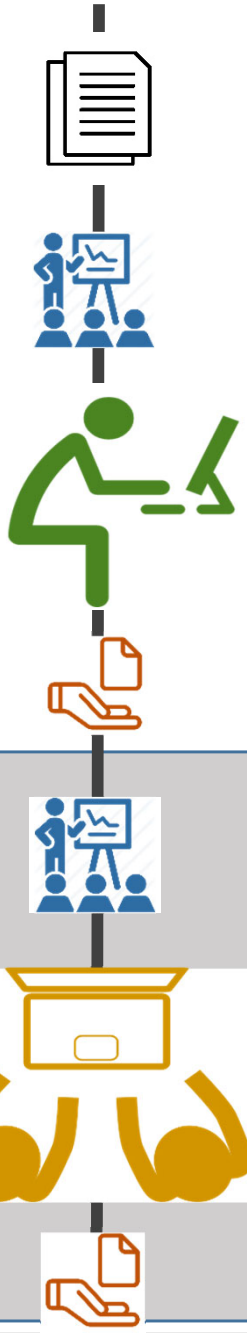
Instructor: Juan Carlos San Miguel (Juancarlos.Sanmiguel@uam.es)

PÁZMÁNY PÉTER CATHOLIC UNIVERSITY

UAM
Universidad Autónoma de Madrid

université de BORDEAUX

TIMELINE



05/05/22 — Materials available in Moodle

05/05/22 — Presentation of programming assignment LAB3
Description of individual work

05/05/22 - 13/05/22 — Individual work

13/05/22 — Submission of individual work

13/05/22 — Description of collaborative work available
Moderated Q&A session for individual work
(individual session, requested by teacher if needed)

13/05/22 - 27/05/22 — Collaborative work

27/05/22 — Submission of collaborative work

EVALUATION

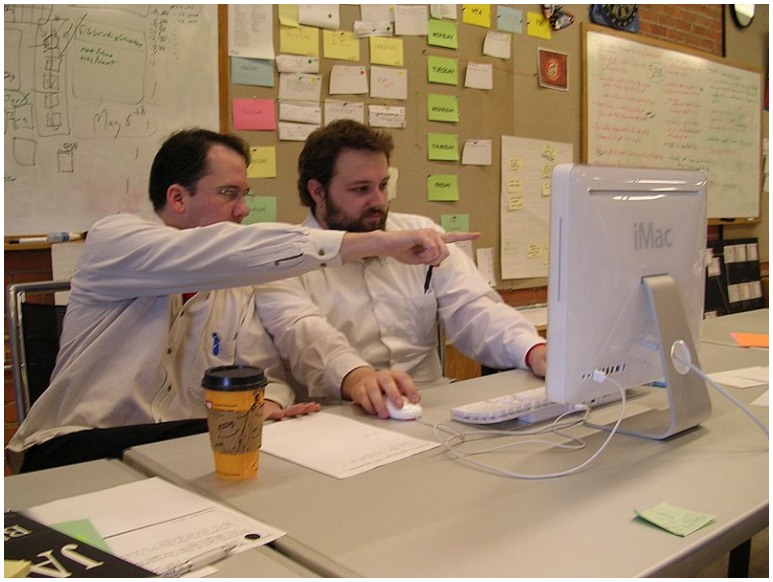
- This lab assignment will be **graded with 10 points**

Type	Concept evaluated	TASK	Max. Score	Criteria evaluated
Individual (35%)	Source Code	Task 4.1	2.5	Code: Functional requirements (60%) Code: Design & structure & Style (40%)
	Report	Task 4.1	1	Report: Introduction & methods (40%) Report: Analysis & discussion (60%)
Collaborative (65%)	**Source Code**	**Task 4.2**	**2**	**Code: Functional requirements (30%) Code: Design & structure & Style (40%) Report: Introduction & methods (30%)**
	Report	**Task 4.3**	**2**	**Report: Experimental methodology (40%) Report: Analysis & discussion (60%)**
		Task 4.4	**2**	**Report: Experimental methodology (40%) Report: Analysis & discussion (60%)**
		Task 4.5	**0.5**	**Report: Analysis & discussion of team dynamics. Teamwork log. (100%)**
		TOTAL	**10**	

- Delivery not following requirements: -0.5 points
  (e.g., submission without makefile or without required directories; report not using the course format)
- Late delivery after the remaining days of the late policy (remember 4-days in total)
  - -25% (one day), -50% (two days), -75% (three days), -100%(>= four days)

COLLABORATION

- Methodology based on pair programming[1]
  - Two programmers/students do a combined effort to develop software using the same computer (either remotely or physically in the same place)
  - Roles:
    - *Driver:* develops and writes code, controls the inputs (mouse, keyboard)
    - *Navigator:* checks written code, suggests alternatives, looks for errors, asks for clarifications,…
    - Roles are exchanged periodically



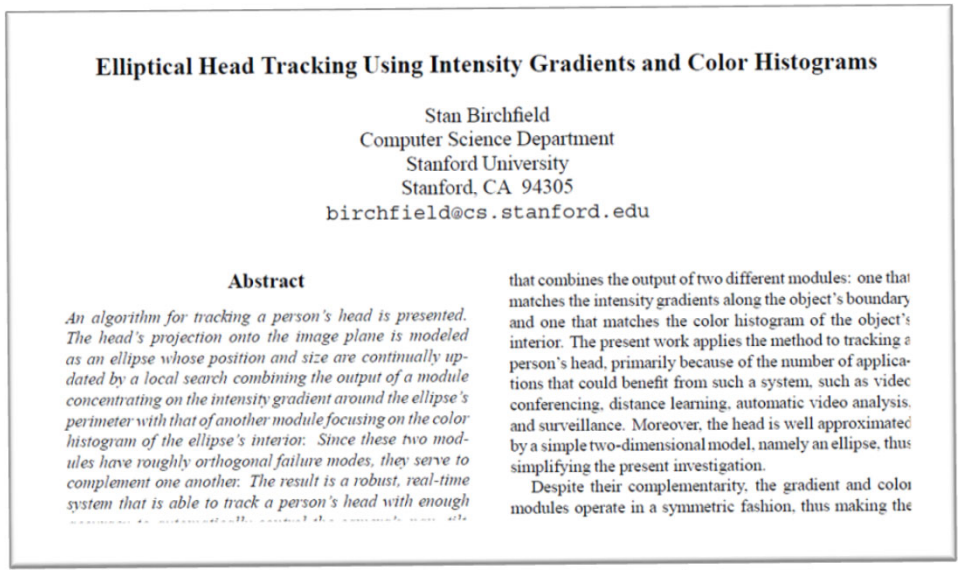
Source: Wikimedia.commons.org

  - Methodology frequently applied in the software and tech industry and (to a lesser extent) for university teaching in programming courses
  - Further documentation and guidelines are provided in Moodle

[1]Bevan, J. et al. “Guidelines for the use of pair programming in a freshman programming class”. In IEEE CSEE&T, 2000.

4.2 Tracker based on color and gradients: code implementation

Objective: implement the tracker described in the paper "Elliptical Head Tracking Using Intensity Gradients and Color Histograms"



Elliptical Head Tracking Using Intensity Gradients and Color Histograms

Stan Birchfield
Computer Science Department
Stanford University
Stanford, CA 94305
birchfield@cs.stanford.edu

Abstract

An algorithm for tracking a person's head is presented. The head's projection onto the image plane is modeled as an ellipse whose position and size are continually updated by a local search combining the output of a module concentrating on the intensity gradient around the ellipse's perimeter with that of another module focusing on the color histogram of the ellipse's interior. Since these two modules have roughly orthogonal failure modes, they serve to complement one another. The result is a robust, real-time system that is able to track a person's head with enough

that combines the output of two different modules: one that matches the intensity gradients along the object's boundary and one that matches the color histogram of the object's interior. The present work applies the method to tracking a person's head, primarily because of the number of applications that could benefit from such a system, such as videoconferencing, distance learning, automatic video analysis, and surveillance. Moreover, the head is well approximated by a simple two-dimensional model, namely an ellipse, thus simplifying the present investigation.

Despite their complementarity, the gradient and color modules operate in a symmetric fashion, thus making the

TASK ALMOST COMPLETED WITH INDIVIDUAL WORK,
BUT YOU MUST MERGE INDIVIDUAL CODES OF TASK 4.1a & TASK 4.1b

4.2 Tracker based on color and gradients: code implementation

Objective: implement the tracker described in the paper "Elliptical Head Tracking Using Intensity Gradients and Color Histograms"

Task: create a C++ program using OpenCV 3.4.4 that does visual tracking based on color/gradient features and histogram matching

Test sequences: "bolt1"
(You should get decent results for all features with #bins=16 and #cand=81 for frames 0-150)

Submission:

- Source code with some param settings employed for the test sequence.
- Report: (very)brief overview of the underlaying theory and description of the source code. Describe any implementation choice you may have done different from the paper.

4.2 Tracker based on color and gradients: code implementation

Use the starter code for this task and the codes from 4.1a & 4.1b

Requirements: consider that your tracking algorithm must

- The object model is defined by a histogram using a selected feature
- Initialize an object model by using only the first frame and the groundtruth data
- Generate a number candidate locations using the grid approach described. (Focus only on object centers. Keep fixed the width/height of object location)
- Compare object and candidate models using the Battacharyya distance
- Select the candidate minimizing the Battacharyya distance (Eq1 w/o gradients) (please note that Eq1 maximizes similarity, here we will minimize distance)
- Your algorithm must be able to change the following parameters: model features, number of histograms bins (`#bins`) and number of generated candidates (`#cand`)

Modifications of the original paper:

- Please do not implement the fusion approach in the paper

4.2 Tracker based on color and gradients: code implementation

Suggestions/help:

- Color conversion https://docs.opencv.org/3.4.4/de/d25/imgproc_color_conversions.html
- Histogram creation https://docs.opencv.org/3.4.4/d8/dbc/tutorial_histogram_calculation.html
- Histogram comparison https://docs.opencv.org/3.4.4/d8/dc8/tutorial_histogram_comparison.html
- Suggestions (optional):
  - Use `std::vector` class for accumulation https://bit.ly/395HDRY
  - Use `cv::putText` to plot text over images
  - Use the `ShowManyImages` class to display many images at once (e.g. current frame, candidate image and feature and histogram)
  - Avoid large portions of code in the main function
  - Use/create classes and methods whenever possible
  - Add comments to your code (at least functional units/methods)

TASKS

4.2 Tracker based on color and gradients: code implementation

4.3 Color-based tracking: analysis over real data

Objective: analyze and tune the Color-based tracker to get the max performance in real sequences from https://www.votchallenge.net/

Task: adjust the tracker parameters for each sequence. You must keep fixed other parameters not related to tracking.

Test sequences: “sphere”, “car1”

Submission:

- Source code with the best param settings for “car1”
- Report: experimental methodology, analysis and discussion of results, and a selection of visual examples for good/bad cases. You may use Tables to compare the different parameter settings.

4.2 Tracker based on color and gradients: code implementation

4.3 Color-based tracking: analysis over real data

Suggestions:

1) Analyze the tracking problems that exist for test sequences

2) Starting from the code developed in task 4.2, assume 16 bins for computing the histograms and explore the effect in tracking performance (time and accuracy)…

a) …of the different color features implemented (e.g. Gray,H,S,R,G,B)

b) …of the changes in the parameter #cand (e.g. higher/lower than 100)

(To get the max grade, you do not need to test all possible variations for all sequences, provide a subset of reasonable experiments and solid conclusions over these experiments)

3) Any other parameter testing you may want try (check first with lecturer)

Performance evaluation metrics are needed to be computed in order to compare the different parameter or feature settings.

TASKS

4.2 Tracker based on color and gradients: code implementation

4.3 Color-based tracking: analysis over real data

4.4 Gradient-based tracking: analysis over real data

Objective: analyze and tune the HOG-based tracker to get the max performance in real sequences from https://www.votchallenge.net/

Task: adjust the tracker parameters for each sequence. You must keep fixed other parameters not related to tracking.

Test sequences: "basketball","ball2"

Submission:

- Source code with the best param settings for "basketball"
- Report: experimental methodology, analysis and discussion of results, and a selection of visual examples for good/bad cases. You may use Tables to compare the different parameter settings.

4.2 Tracker based on color and gradients: code implementation

4.3 Color-based tracking: analysis over real data

4.4 Gradient-based tracking: analysis over real data

Suggestions:

1) Analyze the tracking problems that exist for test sequences

2) Starting from the code developed in task 4.2, explore the effect in tracking performance (visually and quantitatively)…

a) …of changes in the parameter `#bins` (e.g. higher/lower than 9)

b) …of changes in the parameter `#cand` (e.g. higher/lower than 100)

(To get the max grade, you do not need to test all possible variations for all sequences, provide a subset of reasonable experiments and solid conclusions over them)

3) Any other parameter testing you may want try (check first with lecturer)

Performance evaluation metrics are needed to be computed in order to compare the different parameter settings.

COLLABORATIVE TASKS

4.2 Tracker based on color and gradients: code implementation
4.3 Color-based tracking: analysis over real data
4.4 Gradient-based tracking: analysis over real data
4.5 Reflection on the collaboration and individual work

You must include in the report an appendix with the following information (it does not count towards the max page limit of the report):

1) Difficulties found during the individual work stage
2) Modifications or fixes of individual codes when combining these codes collaboratively for building the pipeline for task 4.2
3) Time log for collaborative work with time/data of each session, the student roles and main objective of the session (keep in mind that it does not need to be very detailed)
4) Overall impression of the hybrid pair programming methodology for LAB4

COLLABORATIVE WORK - WORKLOAD

- Expected workload for each student (~15 hours total):
  - ~0h face-to-face (in classroom)
  - ~15h non-presential

TASK	Expected hours	Type of work
-	0.5 (~5%)	Lecturer presentation of the lab4 – Part II
Task 4.2	4 (10%)	Integration of individual work from task 4.1
Task 4.3	5 (~40%)	Running experiments, adjusting params & Description/analysis in the report
Task 4.4	5 (~40%)	Running experiments, adjusting params & Description/analysis in the report
Task 4.5	0.5 (~5%)	Collaborative log. Reflection on the individual and collaborative work.
TOTAL	15 (100%)	

COLLABORATIVE WORK- SUBMISSION

1. Assignment available on Moodle to submit your material

→ LAB4 – Collaborative work (deadline 27/05/22 10AM)

- The material must be submitted as a ZIP file with the following format *Surname1name1_Surname2name2_lab4.zip*
- The submitted ZIP file will contain
  - Report in PDF format following the template (max 6 pages, task 4.5 does not count for the limit)
  - Three directories "T42", "T43" & "T44" containing :
    - *Makefile* to compile and link the program by simply running *make*
    - "src" directory with all source files (.h, .hpp, .c and .cpp) necessary for compiling and executing the corresponding program in Linux
    - **Please do not submit configuration files of Eclipse**

Each pair must submit the work only once