Welcome to COMP20003

Semester 2, 2020

Kris Ehinger

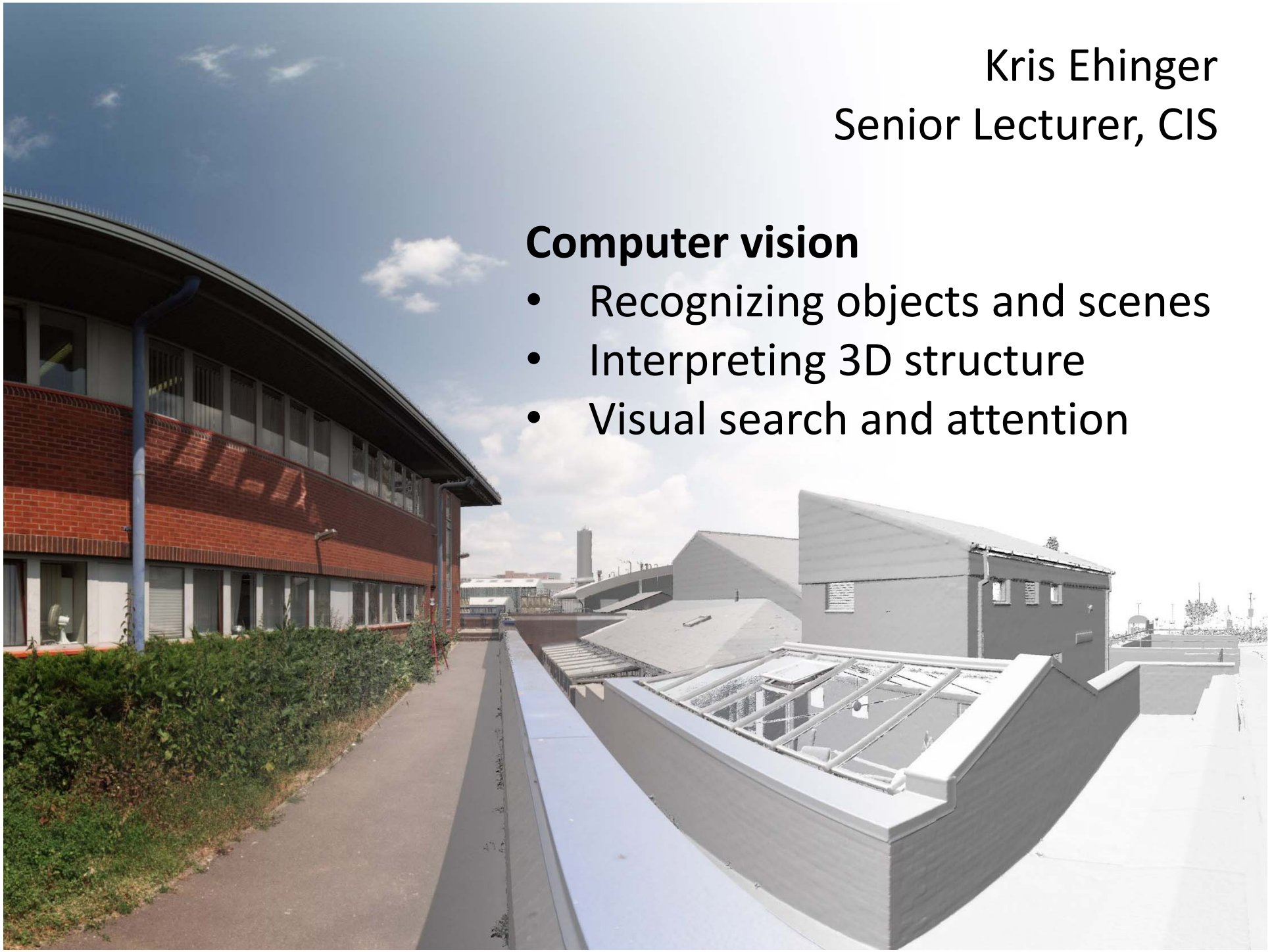
Lecturers

- Nir Lipovetzky (Subject Co-Ordinator)
  - nir.lipovetzky@unimelb.edu.au
- Kris Ehinger
  - kris.Ehinger@unimelb.edu.au

Kris Ehinger
Senior Lecturer, CIS

Computer vision

- Recognizing objects and scenes
- Interpreting 3D structure
- Visual search and attention



Tutors

- Grady Fitzpatrick (head tutor)

- Karl Simon Flores
- Sidakpreet Singh
- Umair Mawani
- Thomas Richard Minuzzo
- Lianglu Pan
- Anh Vo

Contacting us

- General inquiries: Piazza forum on LMS
  - We encourage all students to join in discussions answering other students' questions is one of the best ways to improve your own understanding
  - Please do not post sections of your code or reports publicly on Piazza! If you must include these, private message the instructors
- Personal/private concerns: Email the instructors
  - If you email us about a general inquiry, we may ask you to re post your question in the forum
- Please include COMP20003 in email subject

Lectures

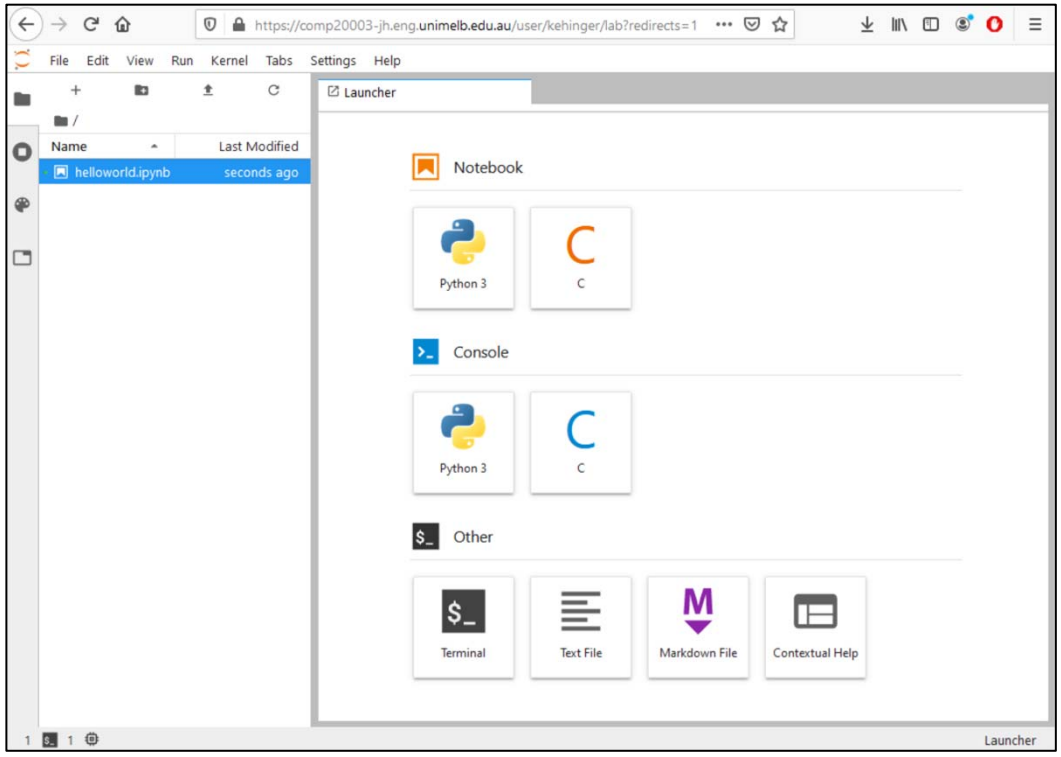
- Each week, we will post a set of pre-recorded video lectures on Monday
- Thursdays 12-1: Live lectures on Zoom
- Thursday lectures will build on the material covered in the pre-recorded lectures

Workshops

- Workshops are 2 hours on Zoom
- Tutorial hour
  - Theoretical concepts behind algorithms and data structures
  - Practical techniques needed to implement these concepts in C
- Demonstration hour
  - Apply theories to practical problems in C
  - Online pair programming

Workshops

- Workshop and assignment material is online
- Jupyter notebooks with a C kernel



Why C?

- Low-level access to memory
- See how algorithms and data structures are implemented in computer architecture
- Great foundation – if you know C, you can learn any programming language

Why gcc, make, gdb, linux, etc.?

- The freedom to run the program as you wish, for any purpose (freedom 0).
- The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help your neighbour (freedom 2).
- The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.

https://www.gnu.org/education/education.html

Assessment

- Assignment 1 (10%, released wk 3, due wk 4)
  - Introduction to C and data structures
- Assignment 2 (15%, released wk 6, due wk 7)
  - Develop an application based on searching/sorting algorithms
- Assignment 3 (15%, released wk 11, due wk 12)
  - Develop an application based on graph algorithms

- Hurdle: 20/40 on continuous assessments

Assessment

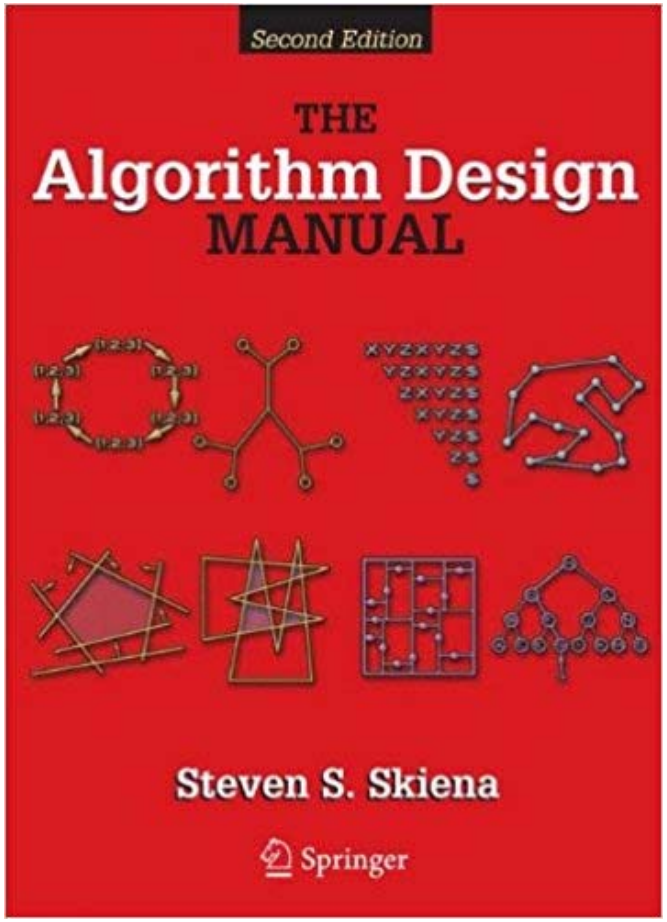
- Mid-semester test (10%, week 8)
  - Expected format: Online, < 1 hour
- Final examination (50%, end of semester)
  - Expected format: Online, 3 hours
- Hurdle: 30/60 on test+final exam

Subject material

- LMS is the primary portal for this subject
- Modules -> Lectures
  - Pre-recorded lecture videos + slide handouts
  - Zoom lecture recording (posted after Thursday)
  - Optional readings and links
  - Self-assessment quiz
- Modules -> Workshops
  - Handouts and code/data
  - Solutions (posted after the last workshop)
  - Workshop recording (posted after the last workshop)

Recommended textbook

- Steven Skiena, *The Algorithm Design Manual*
- Available as an eBook from the MU library:
  - http://library.unimelb.edu.au/ -> Catalogue -> Skiena
  - Note: The copyright license does permit you to download and print for your own personal study.



Additional textbooks

- Other highly recommended books on reserve (ERC High use area):
  - Sedgewick, *Algorithms in C vol 1*, and *Algorithms in C, Part 5: Graphs*
  - Levitin, *Introduction to the Design and Analysis of Algorithms*
  - Cormen, Leiserson, and Rivest, *Algorithms*

What will you learn?

- Introduction to data structures, algorithms, and computational complexity
- For every algorithm:
  - How it works
  - Complexity analysis
  - Implementation
- Basic data structures, and how algorithms interact with them

Algorithms in the real world

- Navigation software: get shortest path to destination.
  - And do it quickly.
- Connect towns or houses to telecommunications network.
  - With the least cost in wire.

Algorithms in the future

- Self-driving cars: trajectory planning, object recognition, localisation, etc.
- Personal assistants: natural language processing, text synthesis, schedule optimisation
- “Big data” prediction algorithms
- Artificial intelligence
- ?