CS 349: User Interfaces

This course is focused on creating user interfaces (UIs), including underlying UI architecture and algorithms, how to implement UIs using frameworks, and theories and methods relevant to interface design.

https://www.student.cs.uwaterloo.ca/~cs349

Jeff Avery & Ed Lank
Winter 2019

User Interfaces

Interface vs. interaction

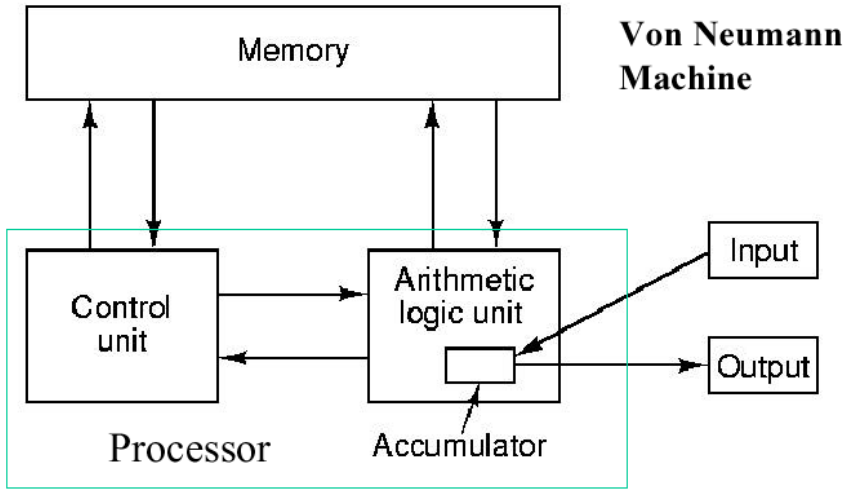
Batch and command-line processing

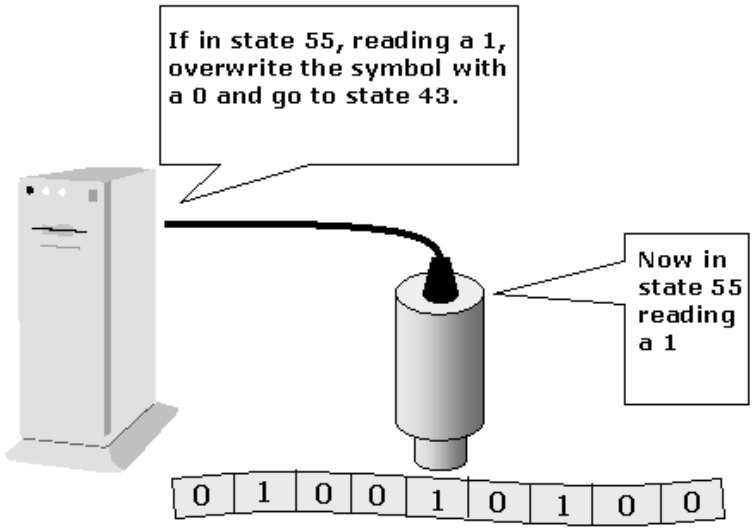
The shift to Graphical User Interfaces (GUI)

Characteristics of GUIs

This might be how we see a “computer”.

We’re experts with this technology, and have a deeper understanding than most users.

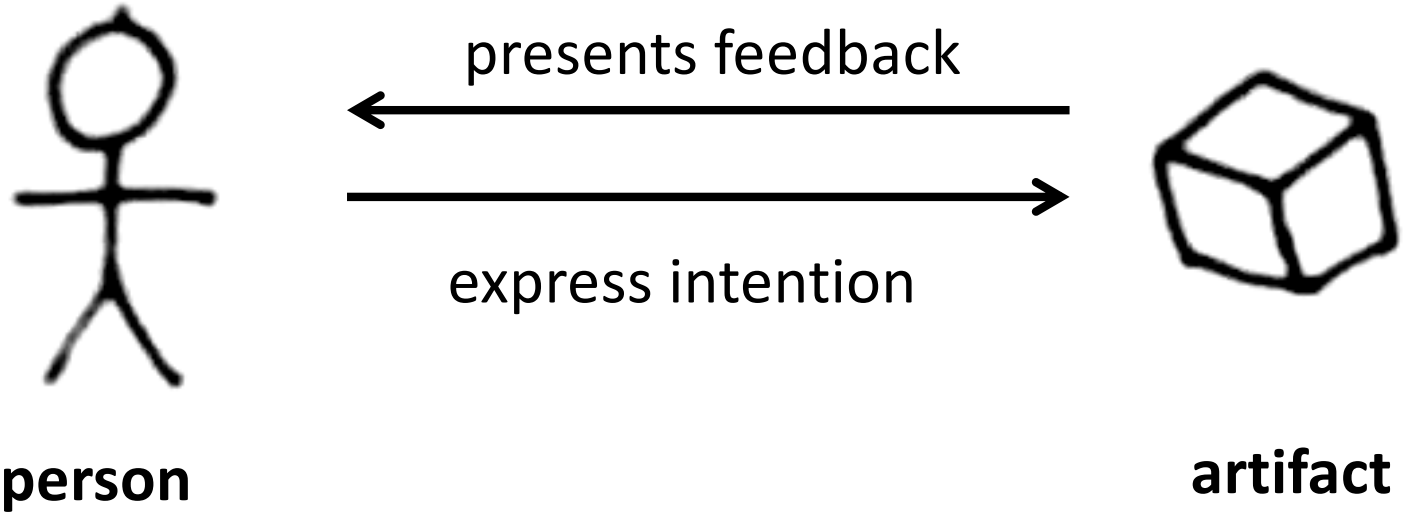




For many users, <u>this</u> is a computer – it's a system they can use to solve a problem, or perform a task.

User Interaction

- The human's view of the computer
- Formally: A user interface is the place where a person expresses intention to an artifact, and the artifact presents feedback to the person.
- We're going to study this interaction cycle in depth.



Interface vs. Interaction

What is the difference between an *interface* and *interaction*?

- **Interaction** refers to actions by user and system over time
  - Interaction is a *dialog* between the user and system
  - Alternates between the user manipulating controls and the system responding with feedback

- **Interface** refers to the external presentation to the user that supports this
  - Controls (what you can manipulate to communicate intent)
  - Feedback (what the program uses to communicate its response)

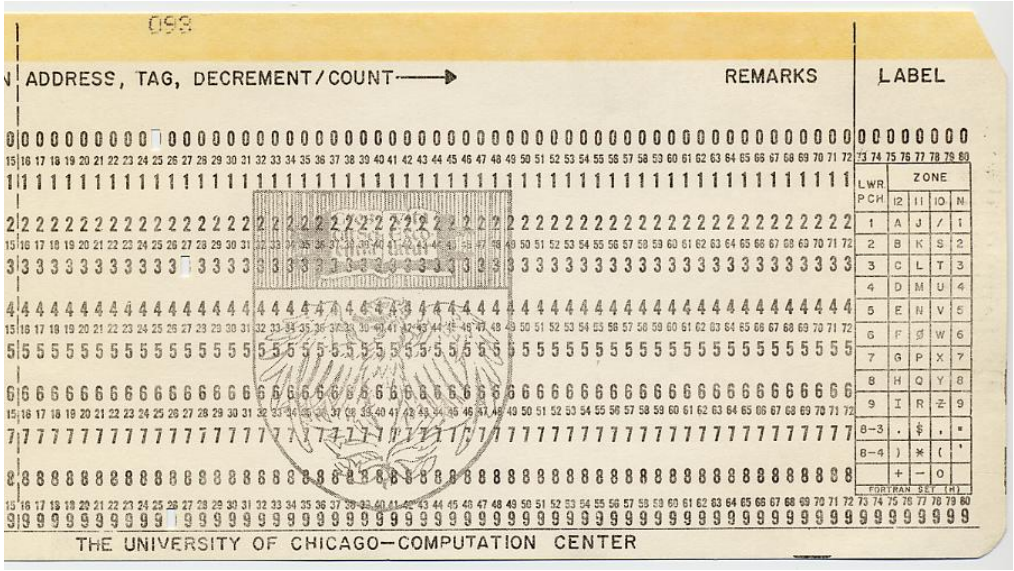
In this course, we're concerned with building efficient graphical interfaces to support interaction.

The History of Interaction…

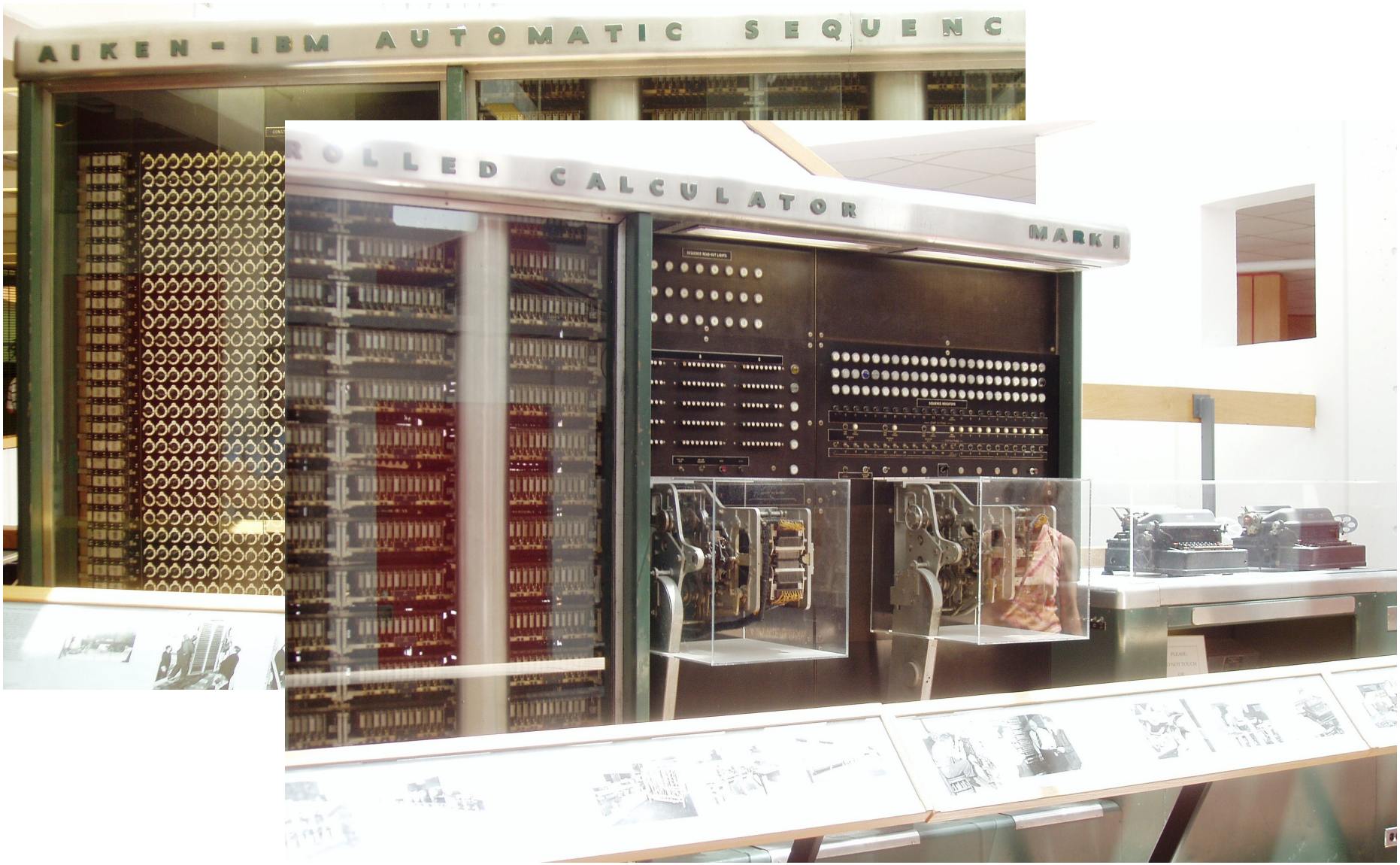
- The history of interaction is the history of making the input and output languages of the machine closer to the input and output language of the user and their tasks
- Interaction has evolved from forms that favoured the machine (when its time was more valuable) to those that favour the user

Batch Interface (1945-1965)

- Interaction style
  - Set of instructions prepared a priori, fed to computer via punch cards, paper tape, magnetic tape
  - Response typically received via paper printout
  - No real interaction possible as system executes instructions
  - Responses received in hours, days
- Users
  - Only used by highly trained individuals

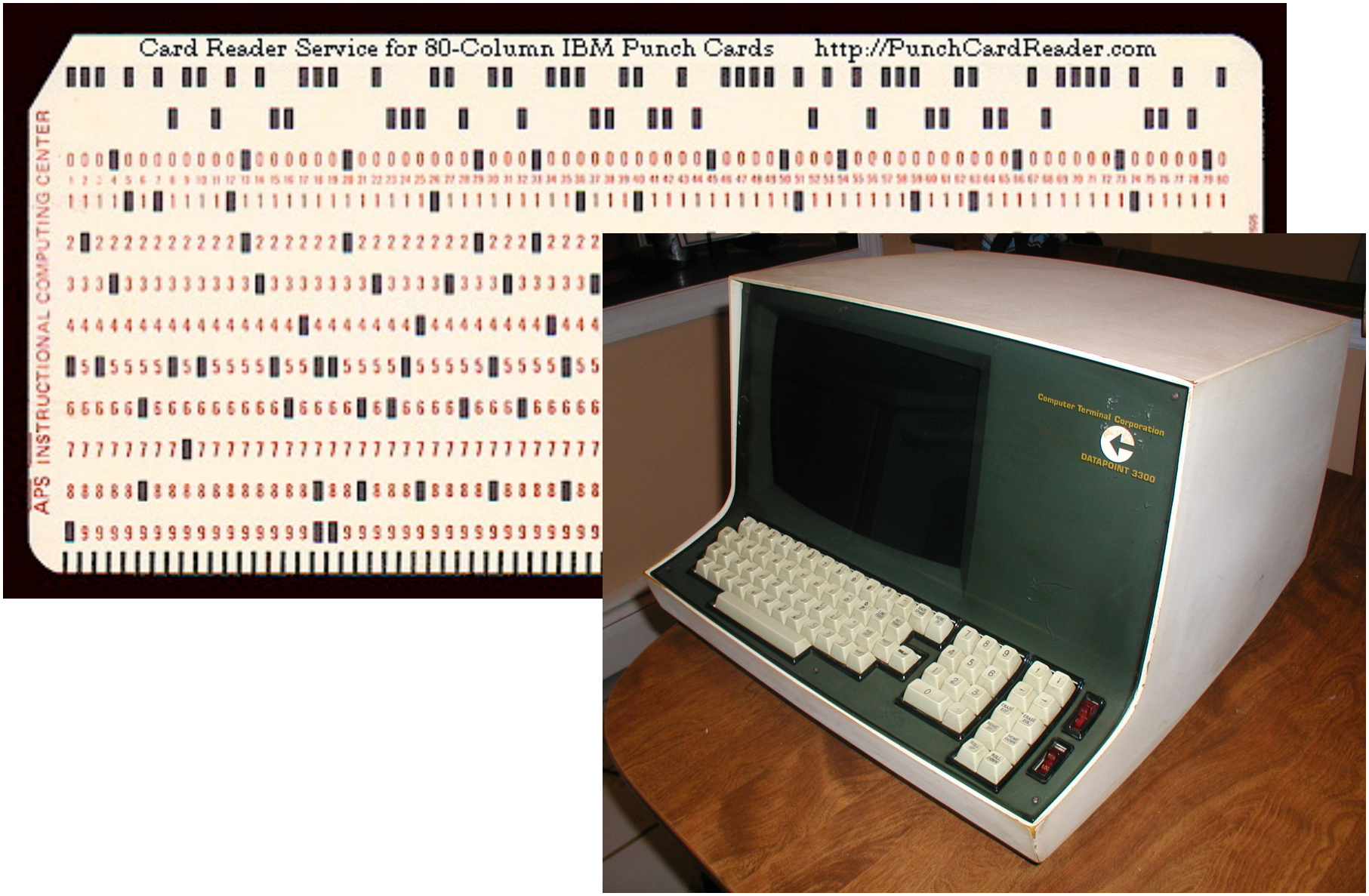


Dials, Knobs, Lights and punch cards (1940s)



Howard Aiken, IBM ASCC / Harvard Mark I

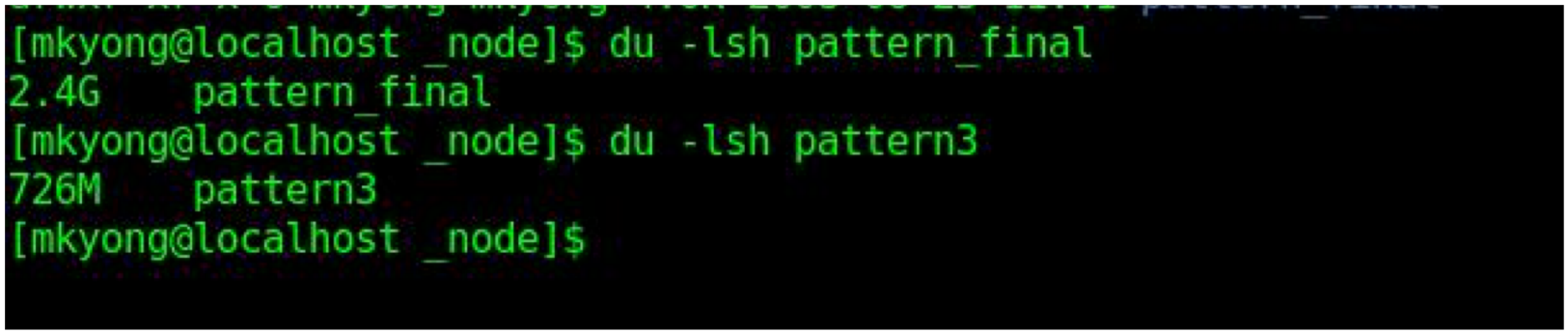
Computing in the 1960s



Conversational Interface (1965 – 1985+)

- Interaction style
  - User issues a command, waits for response
  - Feedback can be given during execution
  - Commands need to be learned
  - Commands are hard to discover
  - Examples: Telephone Voicemail, Bash Shell
- Users

  trained experts



Command-Line Interface

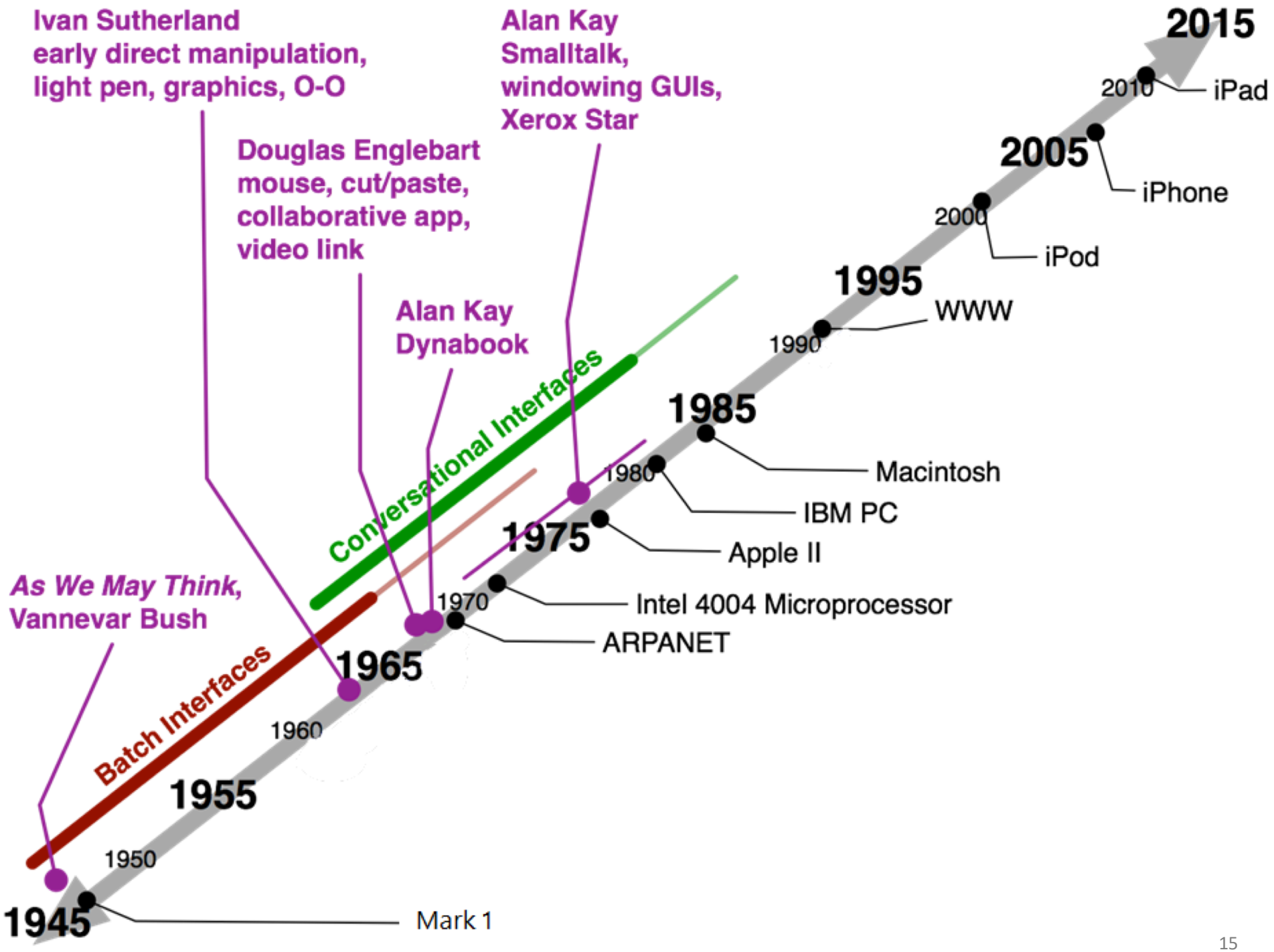
- Advantages
  - Highly flexible: Can combine commands to create sophisticated sets of operations
- Disadvantages
  - Users need to understand the computer
  - I/O is in system language, not task language
  - Requires **Recall** rather than **Recognition**

- Consequences
  - System in control during execution: User cannot refine execution / make modifications during program execution

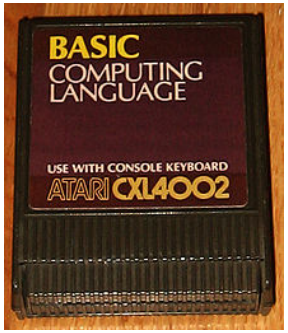
    i.e. you can let the process execute, or cancel, and that's it.

Recognizing User Needs

- Batch and command line interfaces require interaction language closer to the system than task.
- Onus is on user to conform to system i.e. figure out how to structure their problem or task in a way that corresponds to the system.
- Innovations through the 1950s and 1960s directly led to the development of the ”modern computer”, and the graphical user interface.


Ivan Sutherland
early direct manipulation,
light pen, graphics, O-O
Douglas Englebart
mouse, cut/paste,
collaborative app,
video link
Alan Kay
Smalltalk,
windowing GUIs,
Xerox Star
Alan Kay
Dynabook
Conversational Interfaces
Batch Interfaces
As We May Think,
Vannevar Bush
2015
2010
iPad
2005
iPhone
2000
iPod
1995
WWW
1990
1985
Macintosh
1980
IBM PC
1975
Apple II
1970
Intel 4004 Microprocessor
ARPANET
1965
1960
1955
1950
1945
Mark 1

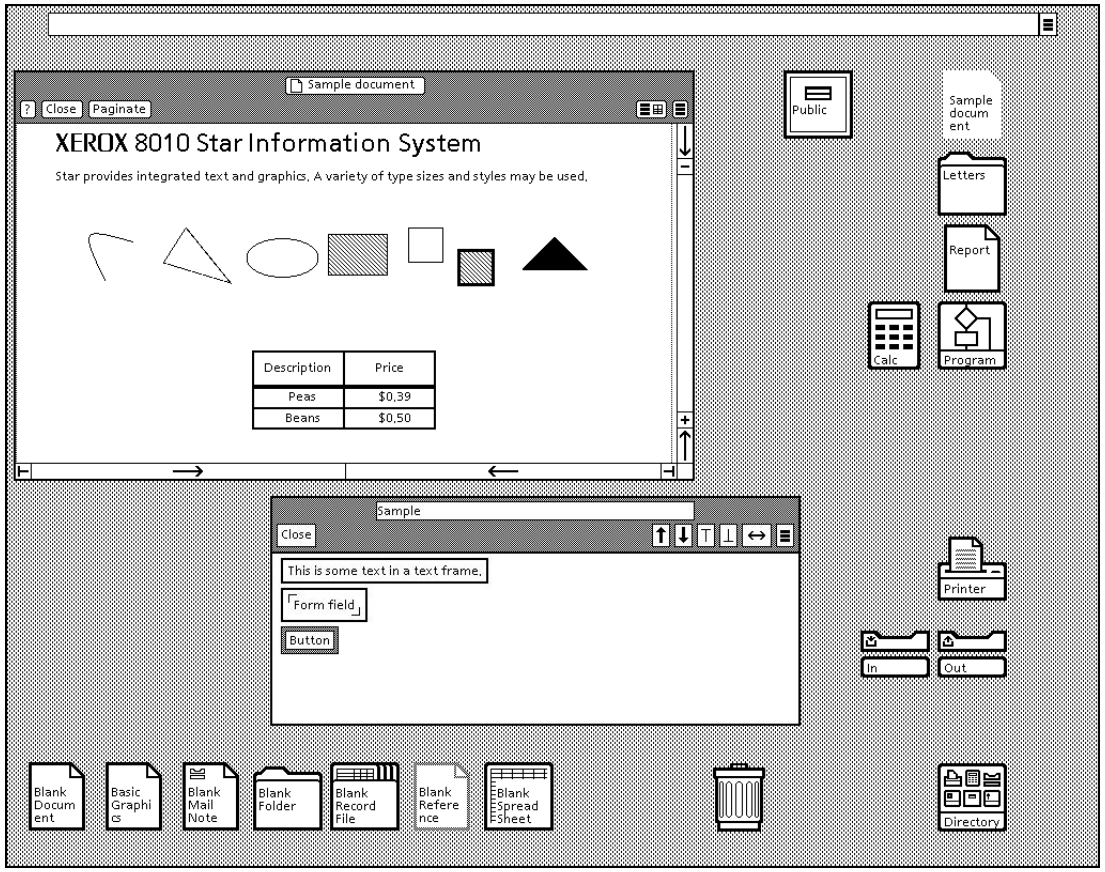
Computing in the 1980s



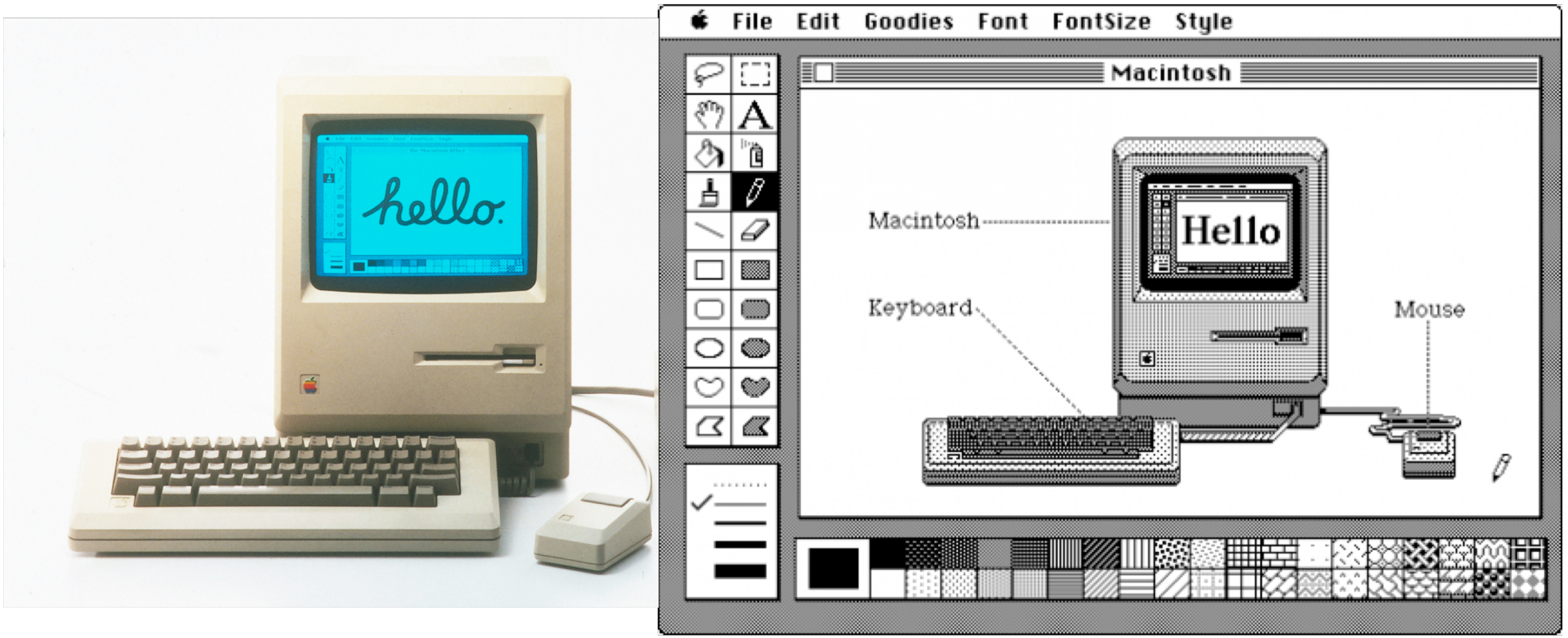

```
READY
LIST

1 REM ********************************
2 REM PROGRAM  : TRAPPING BASIC ERROR
3 REM AUTHOR   : IGOR O.
4 REM PUBLISHER: MOJ MIKRO MAGAZINE
5 REM ISSUE NO.: 1986, NO.10, PAGE 33
6 REM ********************************
10 TRAP 1000
20 A=100
30 PRINT A/0
999 END
1000 PRINT "ERROR ";PEEK(195);" IN LIN
E ";PEEK(186)+PEEK(187)*256
1010 LIST PEEK(186)+PEEK(187)*256

READY
```

Xerox Star Information System (~1981)

- First commercial computer with GUI
  - windows, icons, folders, mouse, (and Ethernet, file/print servers, email)
  - based on Xerox Alto research ~1974



Apple Macintosh (1984)

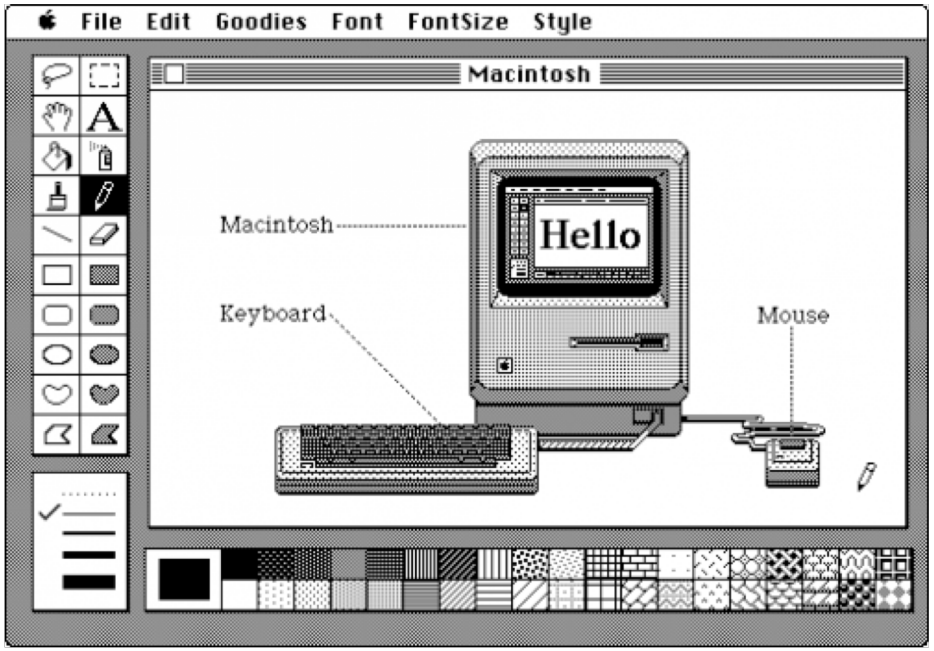


Apple's Macintosh (Jan 1984), brings the GUI to the masses

Windows 95: 1995

Characteristics of a Graphical User Interface (GUI)

- Hardware interface
  - High resolution, high refresh graphics display
  - Keyboard
  - Pointing device (e.g. mouse)
- Software interface
  - supports graphical input and output



A typical GUI design based on the WIMP (Windows, Icons, Menus, and Pointer) paradigm.

This has been the dominant GUI design approach for 30+ years (e.g. Windows, macOS).

Are there other GUI designs?

What are the advantages of a GUI interface?

It enforces a very particular interaction style.

1. The **user remains in control** at all times
   - The system waits for input, then responds
2. It emphasizes **recognition** of interface features over **recall** of commands
   - Enables discovery of options and experimentation
3. It uses **metaphor** to make the interface more familiar
   - Graphical objects results in an interaction language that is closer to users' own language, and closer to the task domain
   - e.g. "desktop", "folder", "drag-and-drop",…
   - What does this mean and what are its consequences for interaction?

These factors taken together make GUI interaction is more approachable and opens the interface up to a broader audience

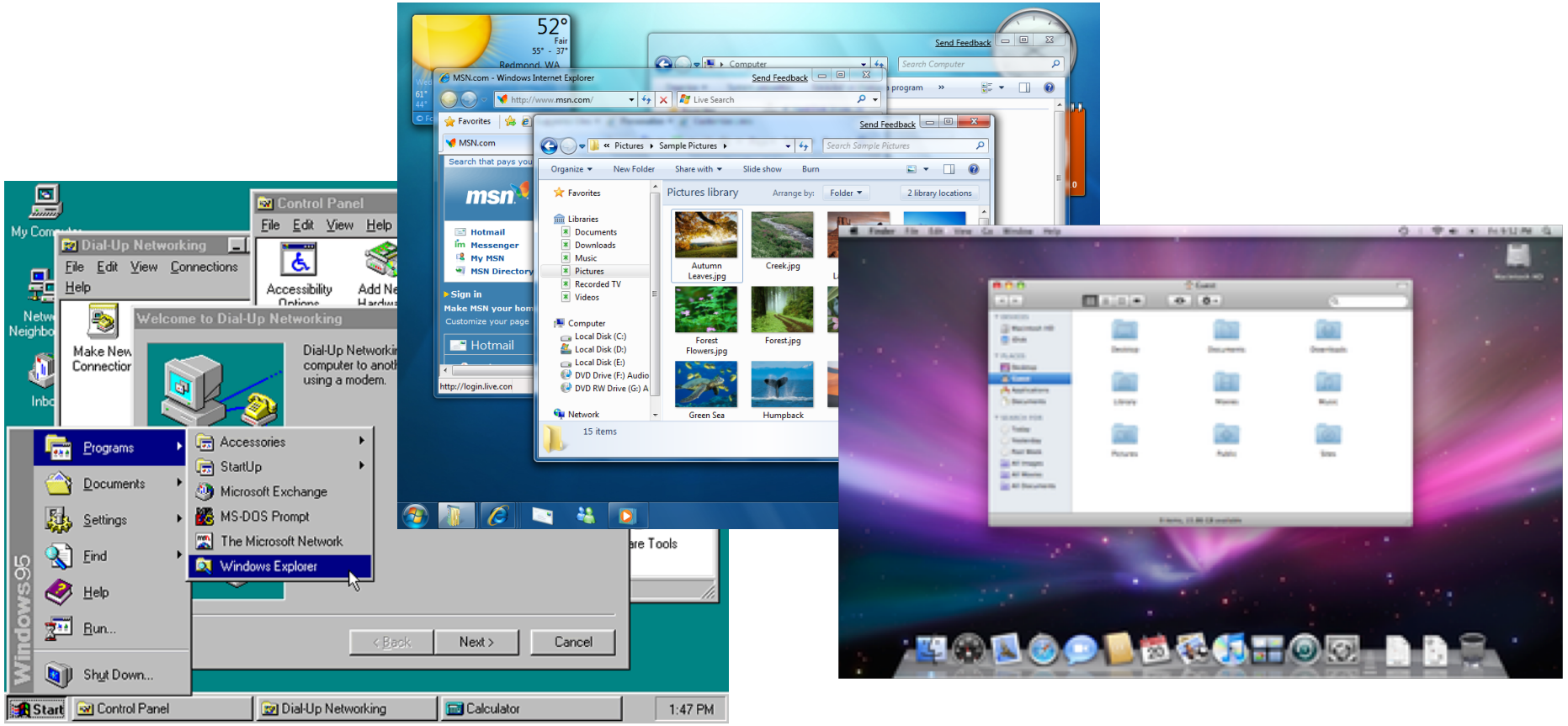
Computing Today

Modern GUI

Interfaces remain graphical and based on the original GUI paradigms.

A GUI does not have to look like Windows or macOS, or rely on mouse/KB

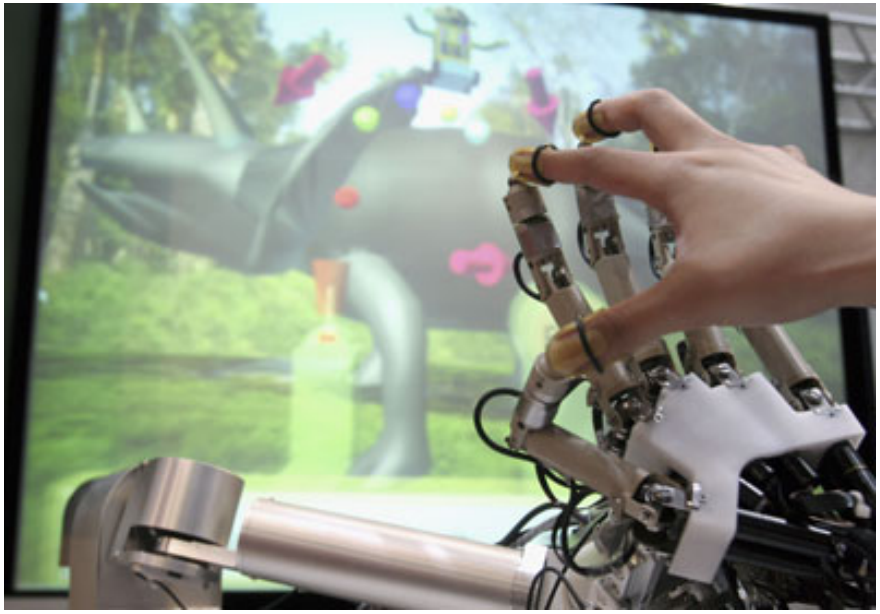
- WIMP (Windows, Icons, Menus, Pointers) is a familiar paradigm



Touch interfaces

Smartwatch

Apple iPhone



Haptic interface

Multitouch wall

Gestural interface

“Kinect” (2010)

“MYO armband” (2013)

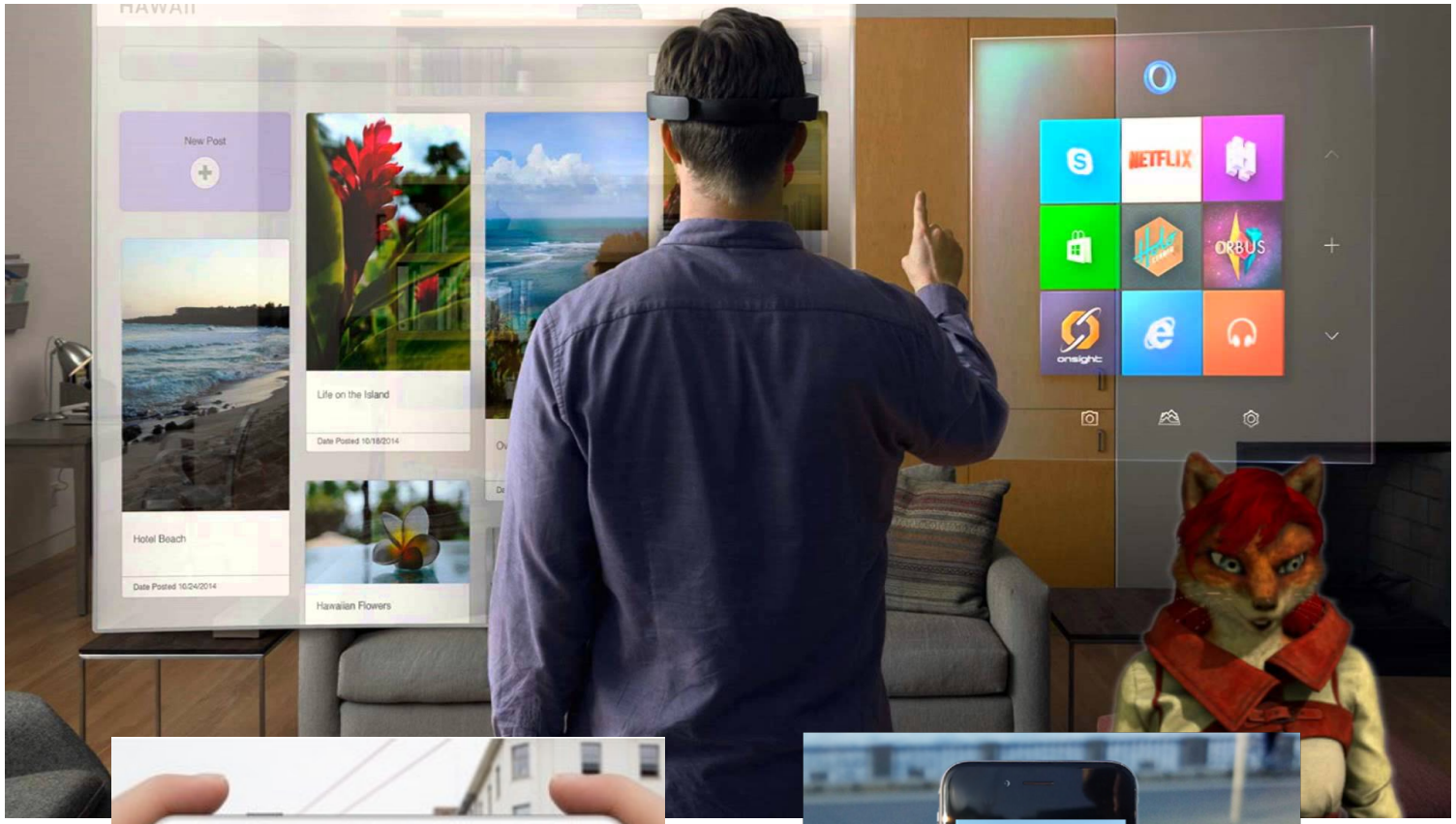
“Leap Motion” (2012)

Voice interface

Apple HomePod Google Home Amazon Echo

Augmented Reality

- Microsoft HoloLens



Why Study HCI?

- Well designed interfaces empower people to do things they couldn't otherwise do
  - Movie production, music production, image editing, assistive technologies, …
- A well designed tool can change the world
  - The web browser, Linux, iPhone, spreadsheet, email, instant messaging, git, live streaming, …
  - Smartphones, tablets (multi-touch)
  - Voice agents (voice)
- For many people, the UI is the computer.
  - Building effective UIs makes computers compelling and useful for many, many people

Syllabus

What to expect

How to be successful

Next steps

Goal

- Our overall objective is to teach you to build compelling and useful user-interfaces, across a variety of platforms and devices.
- The focus of this course is on *building* user-interfaces.
  - User-Centered Design (UCD) is covered more extensively in CS 449.

Learning Objectives

- Understand the architecture, algorithms and design principles underlying common user-interfaces (and UI frameworks)
- Demonstrate the ability to implement a compelling and useful UI on both desktop and mobile platforms.
- Articulate and use basic theories and methods for UI design.
- Leverage Human-Computer Interaction research directly related to building user-interfaces.

Lectures

- Lectures will consist of slides, code demos.
  - Slides and demo code will be posted before lectures
  - Not all content will be on the slides.
  - Anything discussed in class is testable (unless stated otherwise)

    https://www.student.cs.uwaterloo.ca/~cs349

- Laptops and tablets should be used to:
  - View lecture slides
  - Take your own notes
  - Look up content related to lecture
  - Try sample code

- During lecture, please do not:
  - Discuss sports loudly with your friends
  - Work on assignments (esp for other classes)
  - Email, text, IM, Facebook, tweet, tumble, snapchat, instagram, tinder…

Evaluation

20% Midterm

40% Final Exam

40% Assignments (5 assignments in total)

- A1 (5%), A2, A2, A4, A5 (10%) each -- *yes, that adds to 45%*

You have to pass each component separately.

- Your assignment average must be 50%+ to pass the course.
- Your weighted exam average must be 50%+ to pass the course.

We will introduce programming languages in-class, but you're expected to (mostly) learn them on your own.

Assignments are your own individual work

- No group assignments!
- We use MOSS automated plagiarism detection

Getting Help

Web Site: www.student.cs.uwaterloo.ca/~cs349/

- Schedule with topics, lecture notes, sample code, recommended reading
- Office hours for lecture and assignment help

Office Hours

- 2 hours of TA office hours every day, usually in MC 3004
- Help with assignments, clarification on course material

Piazza: https://piazza.com/uwaterloo.ca/winter2019/cs349

- Announcements
- Q&A about course material and assignments
- Staff will monitor - best-effort response, but no time-guarantees
- Sign up and *use your real name* (it helps us to help you)
- Answer questions, but don't be *too* explicit - do not post assignment code!