Preparation for the Project

Jonatan Tidare, jonatan.tidare@mdh.se
20211110

The project…

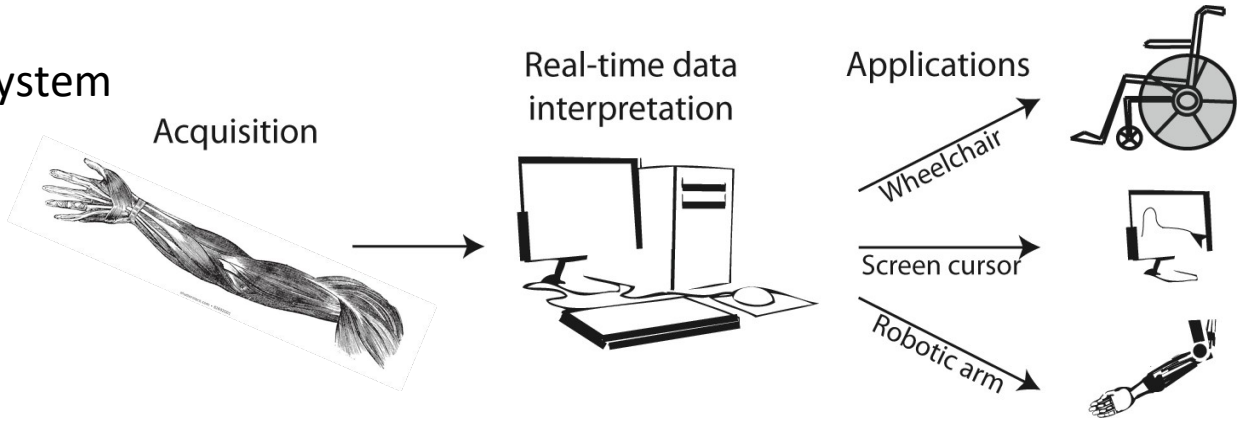
Todays lecture

- Working in a larger project
- Our proposed software
- Prototyping
- Requirements

Working in a larger project

- Top-down?
  - Remember the final deliverable
  - Defines communication end points early
  - Pre-defined functions helps when developing many parts simultaneously
  - Good for System-in-a-loop
  - Requires knowledge about system

We should send EEG data every 50 ms.



Lets assume the start signal is a string.

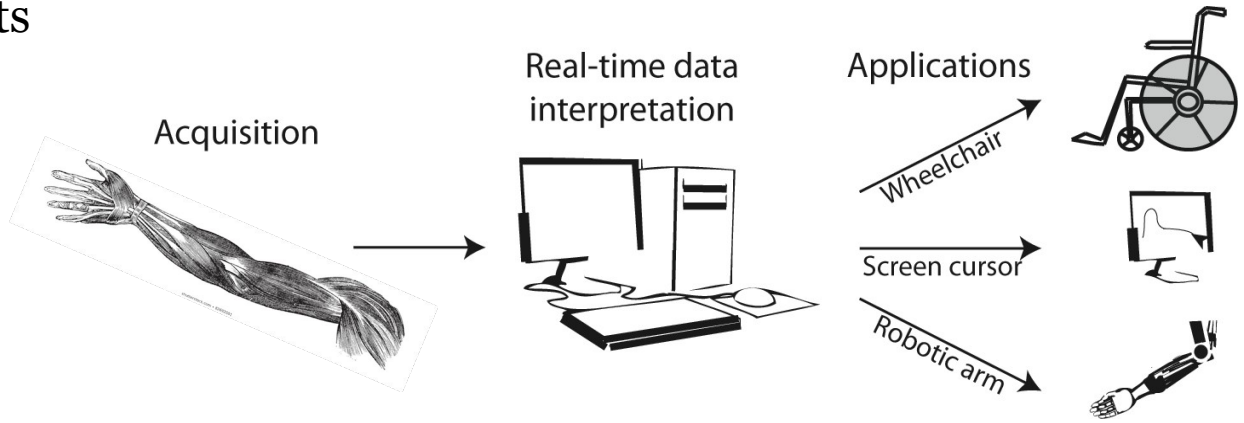
We start with a fix motion on a positive prediction and wait for the hand to return.

Working in a larger project

Top-down

- Can you think of problems arising from these ideas?
- Imagine they happen without communication between components

We should send EEG data every 50 ms.



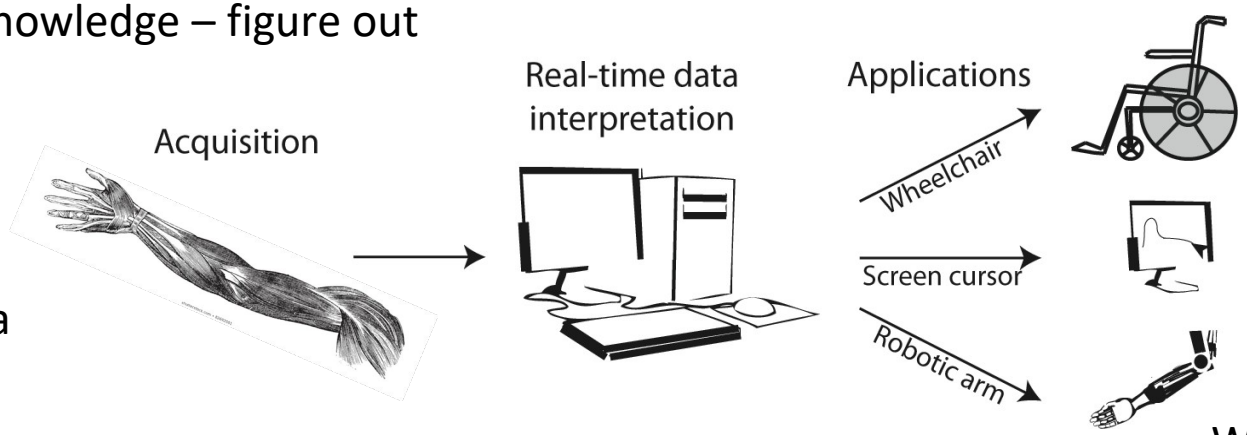
Lets assume the start signal is a string.

We start with a fix motion on a positive prediction and wait for the hand to return.

Working in a larger project

- Bottom-up?
  - Fun to start coding quickly
  - Useful if you lack knowledge on low-level solutions
  - Helps define functions better (but later)
  - May be difficult to assemble project components
  - Lower requirement on knowledge – figure out low-level solutions first

It's actually simpler to always send strings via socket!



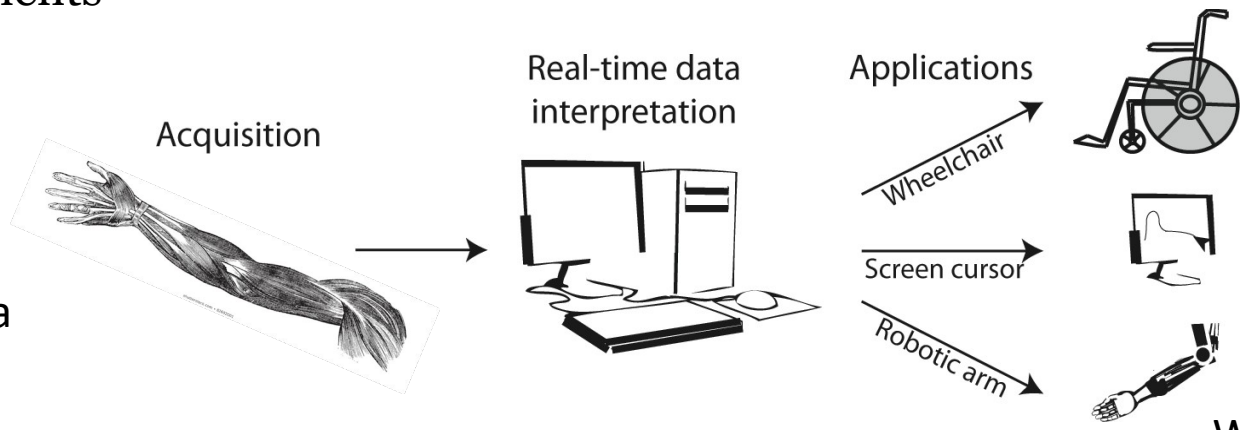
Transforming into fewer features increases classification accuracy!

We could use probabilistic values for motor speed!

Working in a larger project

Bottom-up

- Can you think of problems arising from these ideas?
- Imagine they happen without communication between components



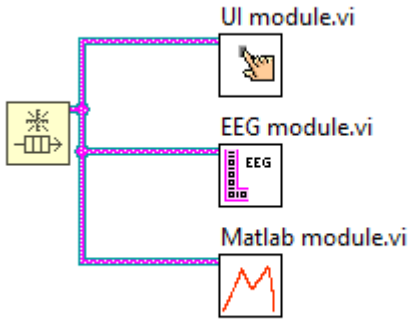
It's actually simpler to always send strings via socket!

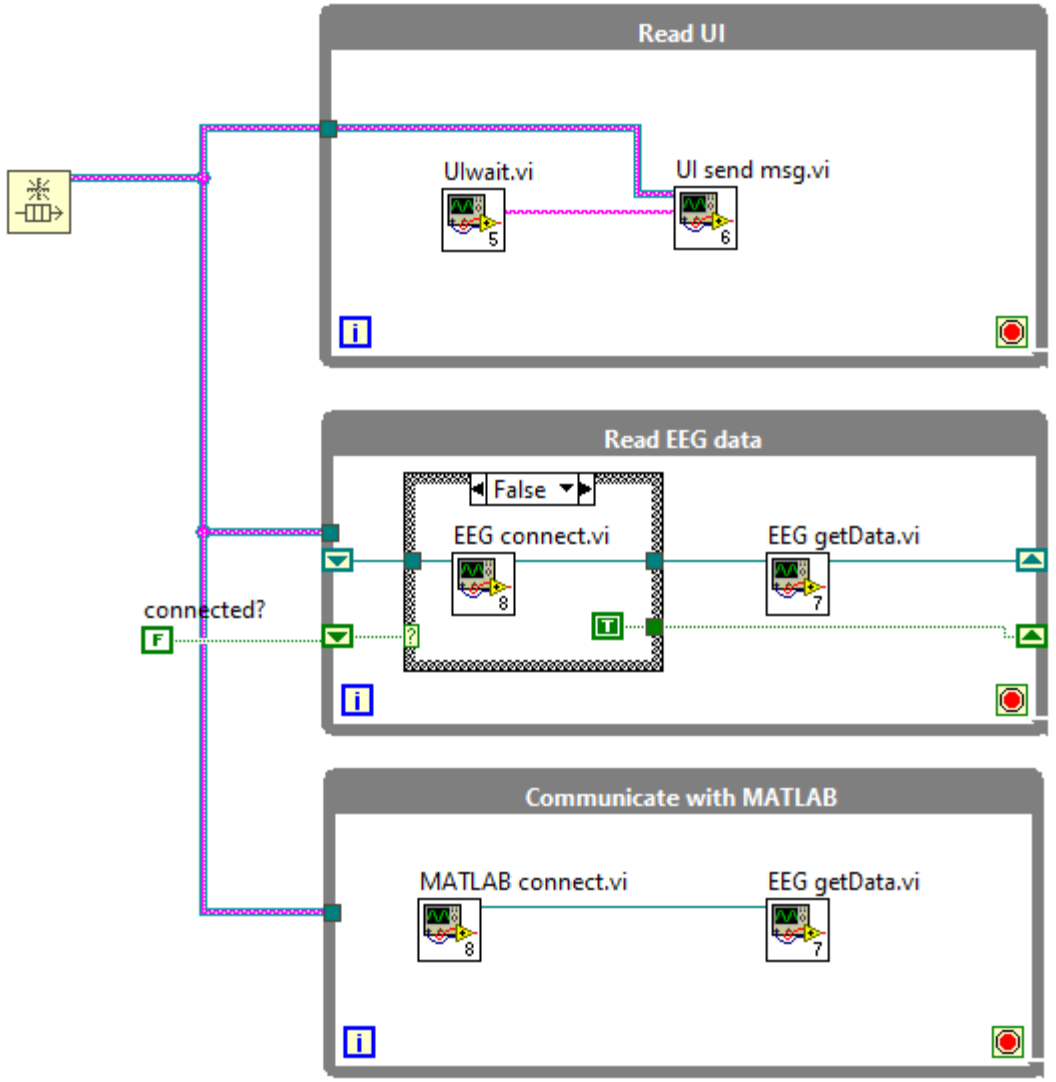
Transforming into fewer features increases classification accuracy!

We could use probabilistic values for motor speed!

Working in a larger project

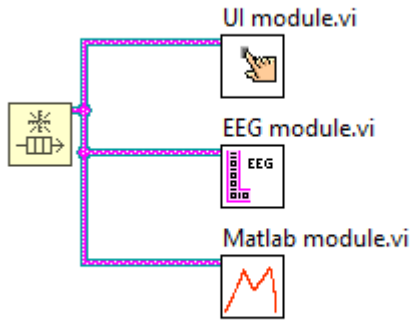
- LabVIEW helps with top-down
- Working top-down automatically gives you a flow chart
- If you develop this early, each component knows the interface to other components
- System-in-a-loop





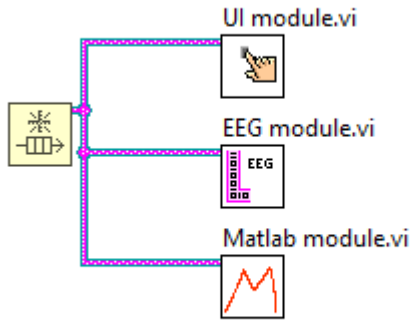
Working in a larger project

- System-in-a-loop
  - Originates from “hardware-in-a-loop”
  - Virtualised HW platform
  - Insert HW parts to evaluate functionality
  - Each component can try their actual code with a functional system
  - Example: ROARy (or unicorn) had 3 components
    - Refuse gathering robot (mdh)
    - Drone surveillence and bin detection
    - Autonomous truck



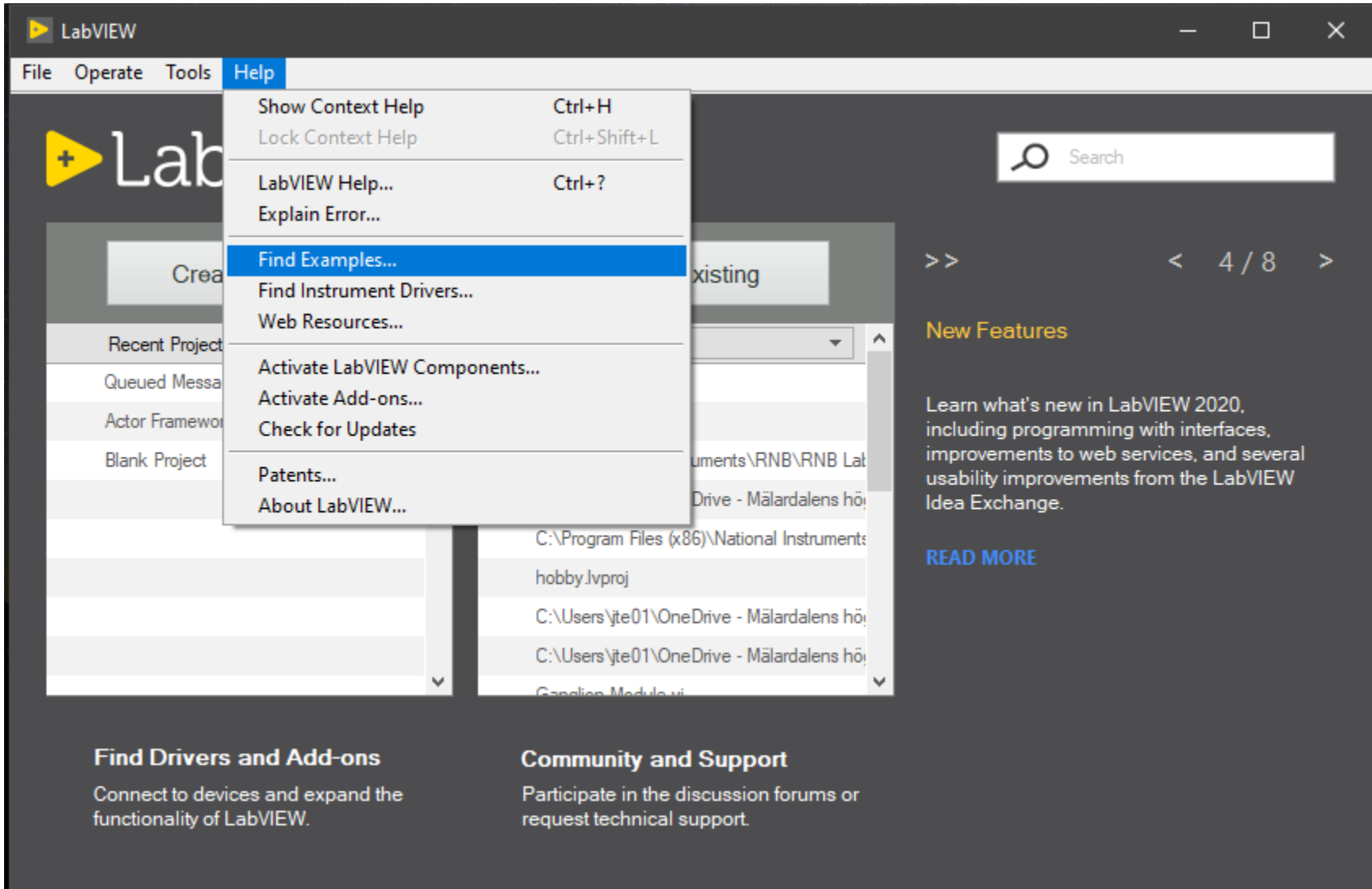
Our proposed software

- LabVIEW
  - Naturally parallelized
  - Lots of templates and example codes
  - If done correctly, becomes its own flowchart
  - Difficult to explore and analyze data with many variables
  - Sometimes graphical language can be messy



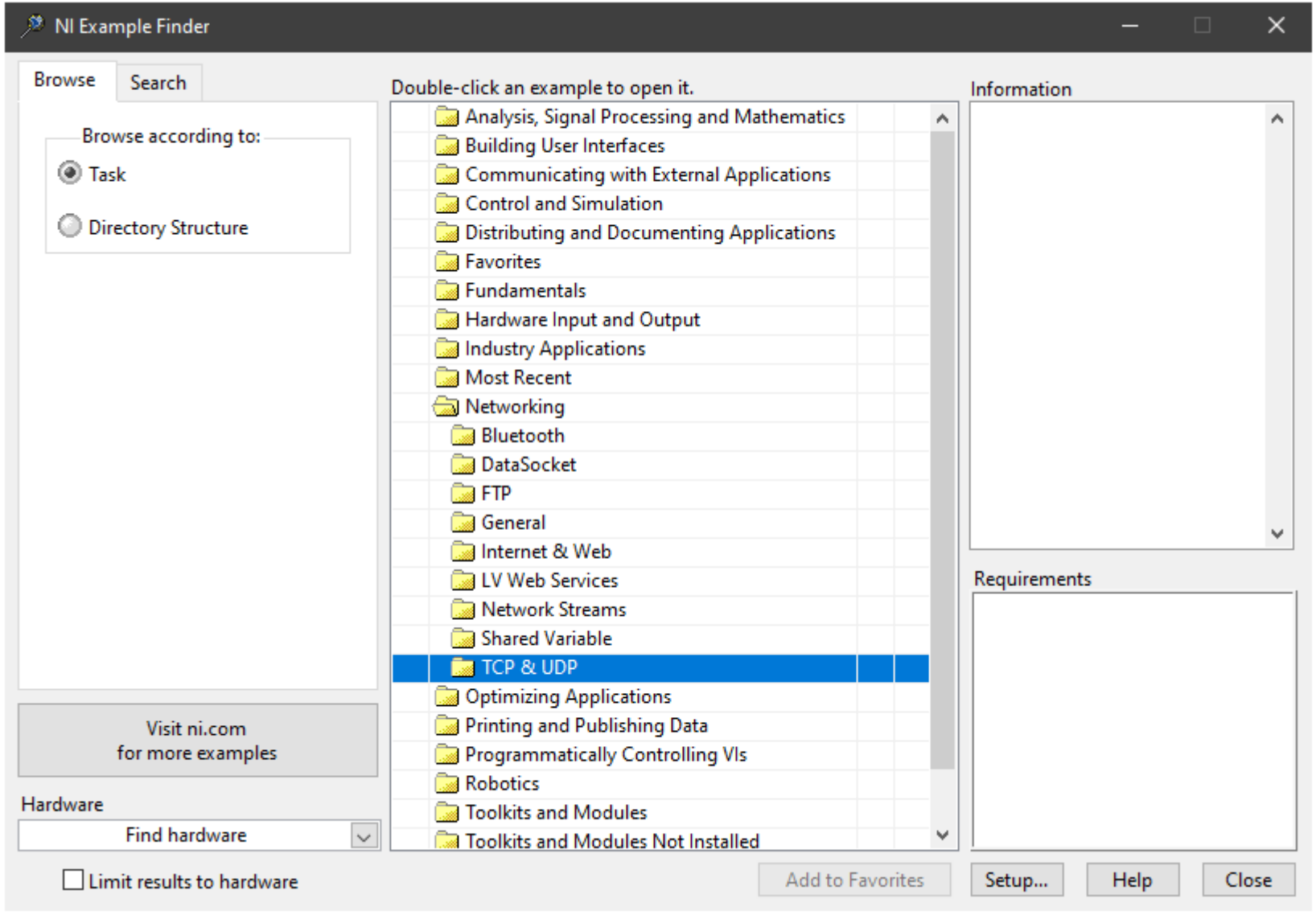
Our proposed software

- LabVIEW
  - Finding example



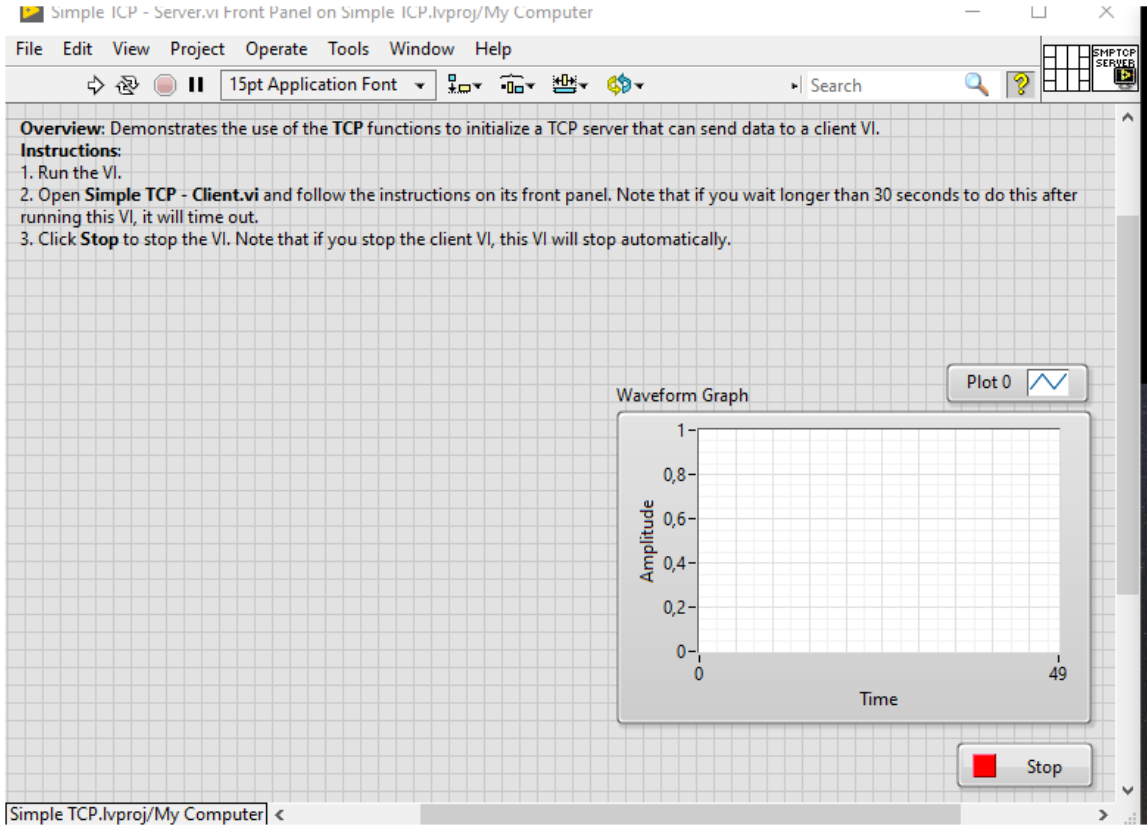
Our proposed software

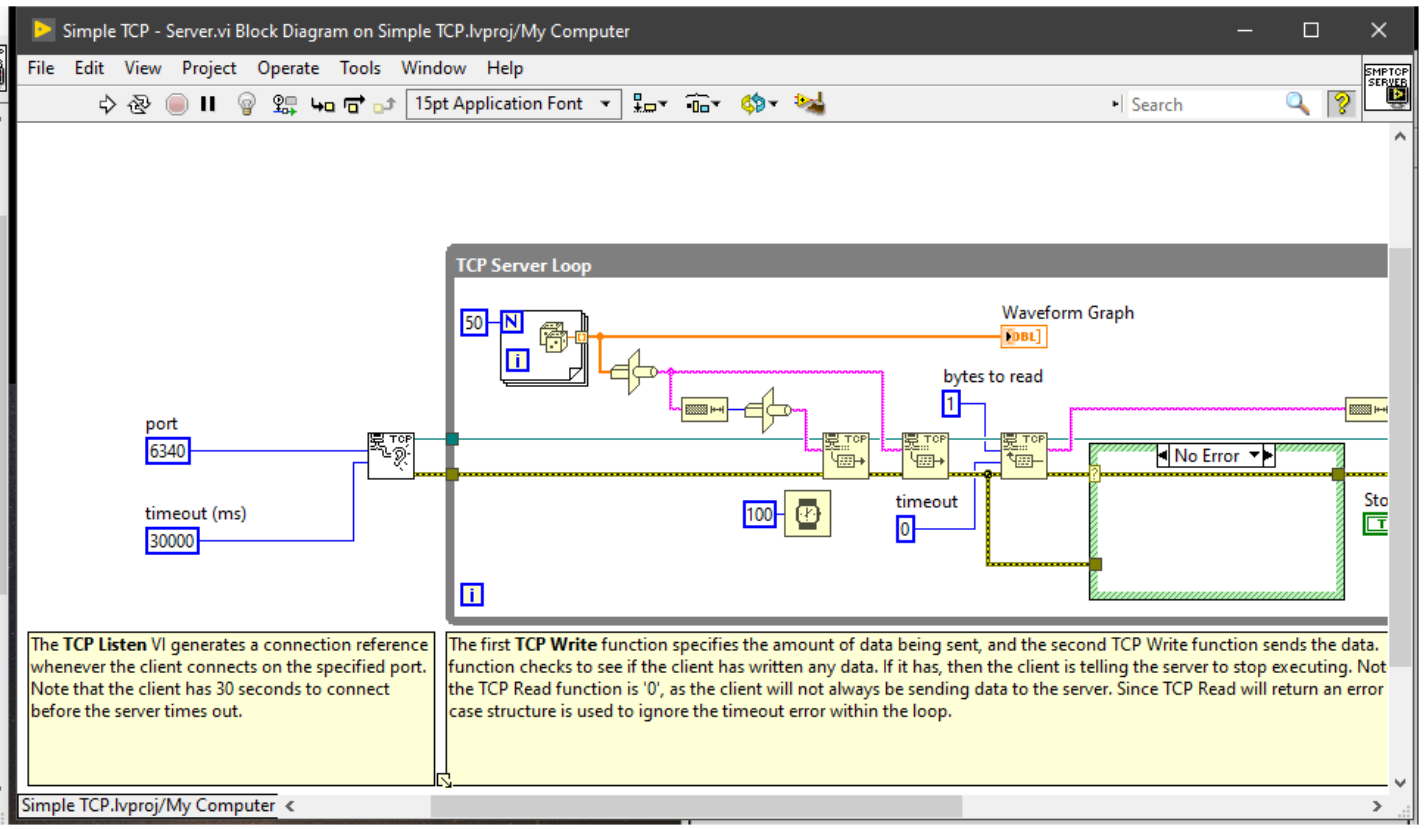
- LabVIEW
  - Finding example



Our proposed software

- LabVIEW
  - Finding example



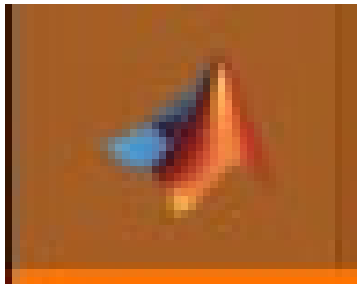


Our proposed software

- LabVIEW
  - Finding example
  - Because a good engineer knows what to look for
  - i.e. “good at googling things”
  - stackoverflow, mathworks, docs.microsoft, ni.com
  - Knows functionality exists, but not necessarily how to code in every language

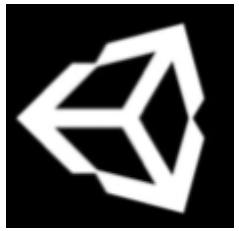
Our proposed software

- MATLAB
  - Single thread, no parallel execution except built-in functions
  - Good for plots and visualization
  - Easy to explore and analyze data
  - Script language – may pause code, change variables, then continue. Incredible for debugging and testing.



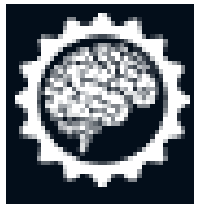
Our proposed software

- Unity
  - Powerful engine for presenting instructions and feedback
  - Many tutorials and sample projects
  - May present images, video, sounds, text etc
  - Written in C# (yet another code language)



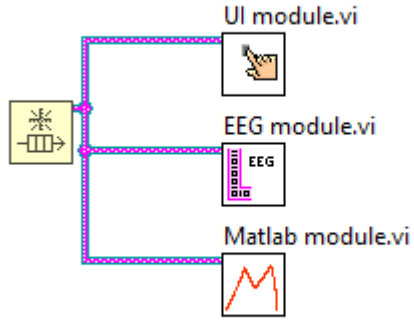
Our proposed software

- OpenBCI
  - Open software for several EEG/EMG recording systems
  - Has GUI but can also stream data
  - More on website (openbci.com/)



Our proposed software

- LabVIEW is required because Comp 3 will program a RoboRio (NI hardware)
- OpenBCI is also required to use the EEG/EMG equipment
- MATLAB is strongly recommended but you may use python/octave or other programs too (let me know if I should install on project laptops)
- Unity is recommended, but both LabVIEW and MATLAB may be used to present instructions on screen (for the protocol).



Prototyping

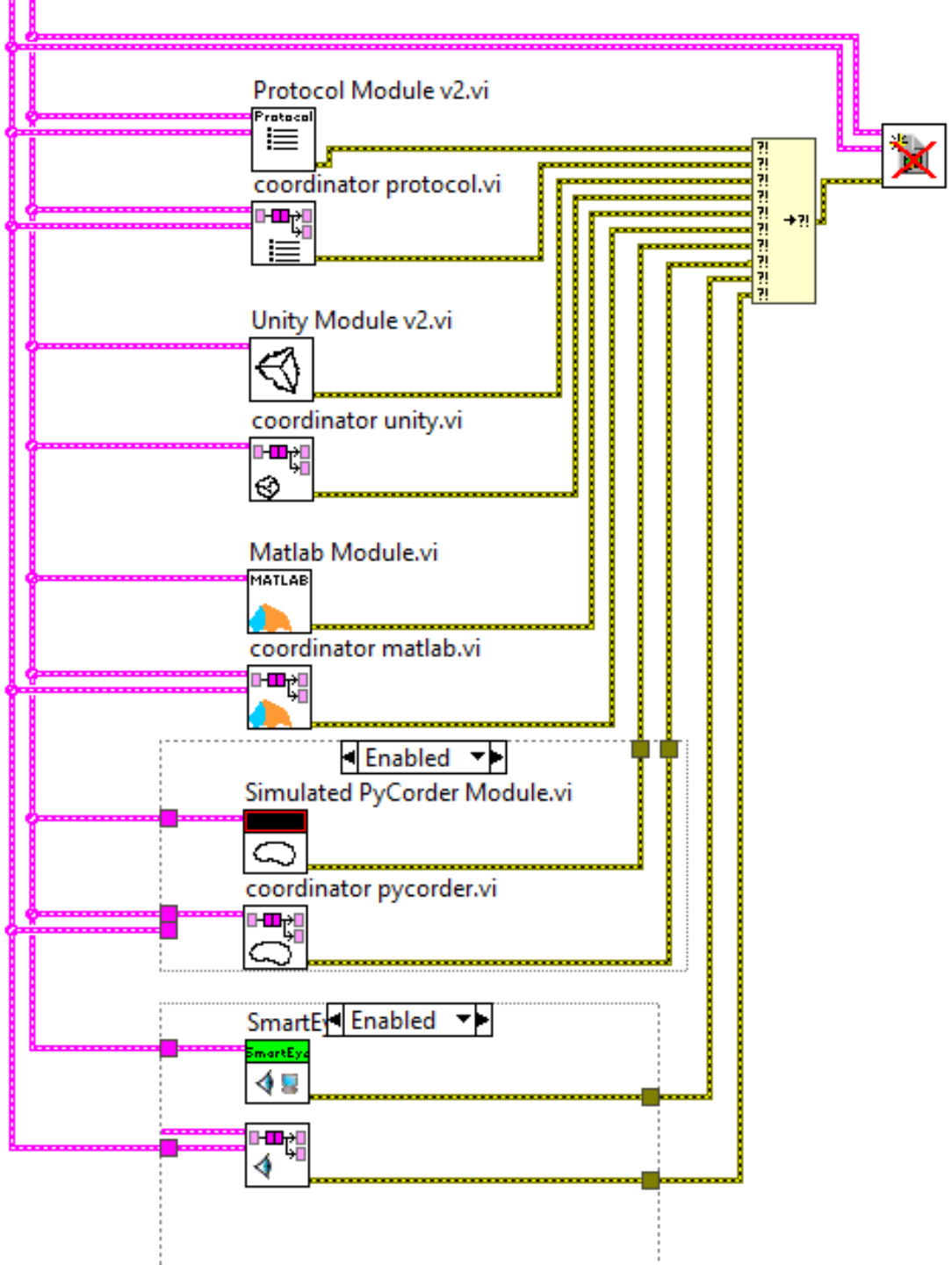
- Important aspects of prototyping
- Proof-of-Concept
- No need for perfection
- Emphasis on working example
- Don’t spend a week optimizing a linked list…

Prototyping

- Proof-of-concept
  - Minimal functionality
  - Re-usability (readability or speed)
  - Bad readability may cost many hours from colleagues or future you
  - Bad optimization may or may not be an issue. An example…

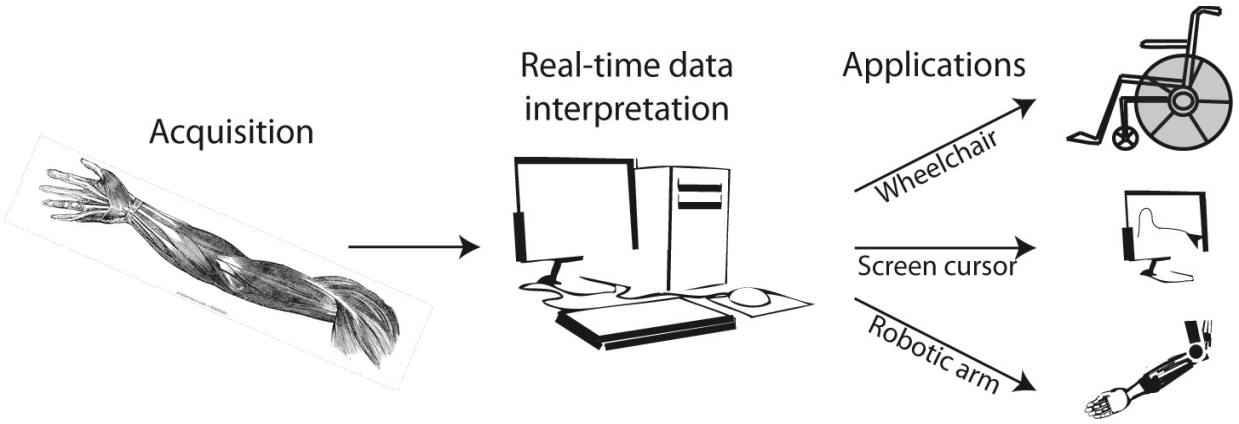
Prototyping

- Proof-of-concept
  - Read or speed (my BCI)
  - One task = one subVI
  - PyCorder module reads data, then sends copies to Matlab Module.
  - Optimize with pointers?
  - Or keep as is to maintain readability. One subVI reads from PyCorder, another subVI sends to MATLAB.
- Where to draw the lines?
  - When the protocol presents an image, unity should update screen
  - When a task begins, MATLAB should start classifying incoming data (or start sending to the end effector)



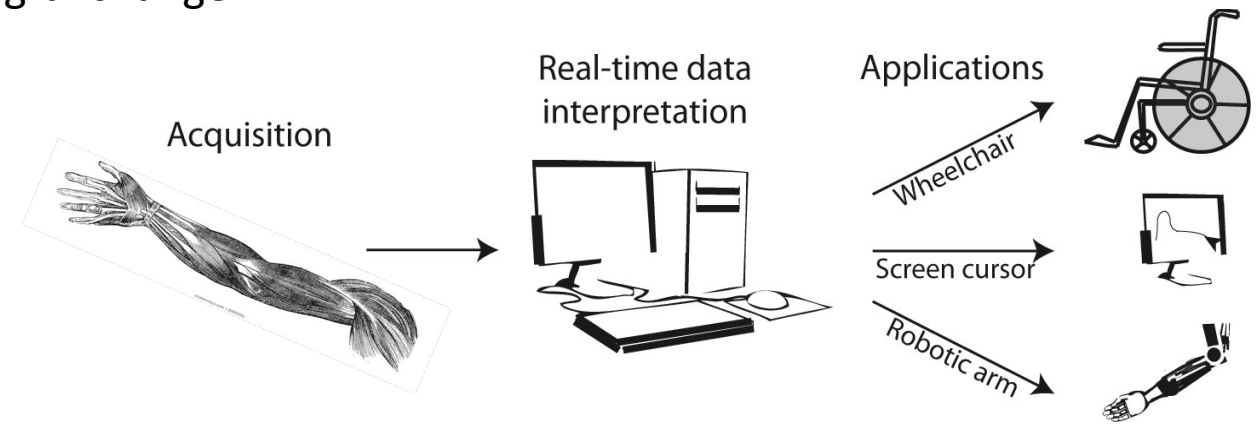
Prototyping

- No need for perfection
- Emphasis on working example
- Requirements
  - My subVIs were fast enough (0.1 – 0.01 ms)
  - Wavelet transform 250 ms
  - You set your own Real-time constraint (in relation to usability of the system)



Requirements

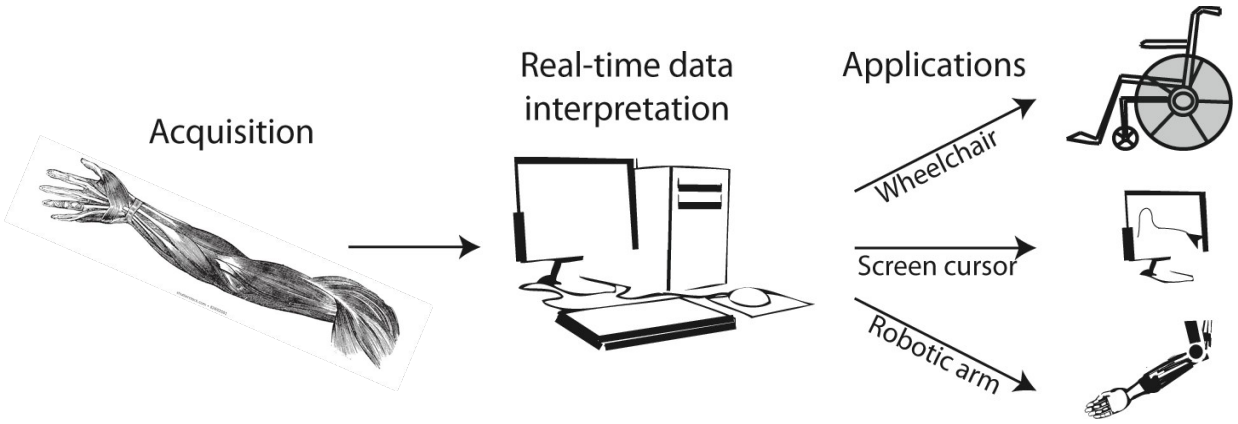
- No need for perfection
- Emphasis on working example
- Requirements
  - My subVIs were fast enough (0.1 – 0.01 ms)
  - Wavelet transform 250 ms
  - You set your own Real-time constraint (in relation to usability of the system)
  - Only optimize if it brings meaningful change



Discussion

The three components correspond to acquisition, analysis and manipulation (end effector). Discuss the following questions in groups!

1. How would you like to work in this project? Consider top-down, system-in-a-loop etc. Do you have experience from other courses/labs?
2. What do you think about the proposed software? Do you agree or oppose the suggestions? What parts/programs do you feel most comfortable with?
3. Have you thought about prototyping as a different skill from other product development? For example, have ever optimized code to be readable rather than efficient?
4. Have you ever optimized code just because? Is it motivated to invest hours even if the optimization isn't required?



End of Lecture 2

Today I talked about
- Working in a larger project
- Our proposed software
- Prototyping
- Requirements

I will upload information about the project and seminars shortly.