Kieran will focus on research into the electronics and control side of the project. This will include how cooperative robotic control systems differ from traditional control systems and the main challenges faced when incorporating multiple robotic devices into one task. Kieran will also research in depth about how other arm designs are controlled, with specific focus on the advantages and disadvantages of a digital twin control system to determine if that form of control systems would be beneficial for this project. Both Jack and Kieran will conduct research into drive systems for example motors, servos, pneumatics etc. Jack will research from a structural and form factor perspective while Kieran will conduct research into the ways each form of drive systems will be controlled.

Plan

* Take relevant notes for all the references that seem useful on humanoid robotics
* Look up all recent papers on actual robots that have been built say in the last 6 years, and note both the designs and performance, in order to critically analyse them
* Review of actuator availability and characteristics…
* Do another 5 robots for a total of 13 robots seams reasonable, but could also look ast summary stuff

Final Hours Plan

* Rewrite intro bullet points - 20 mins
* Write a Summary - 40 min
* Add in a bit more on control to the actuator section - 30 min
* Fix up missing references and other bits and pieces - 30 min
* Add those other bits to the electronics paragraphs - 10 min
* Add some more criticism of sources and identification of gaps - 20 min
* Add in more “birdging” 20 min
* General editing - 45 min
* Add in references - 25 min
* Add pics?
* Change et al to et al.
* Update table of contents
* Submissions and checking - 15 min
  + Add project proposal

Total = 4.16

Out of scope

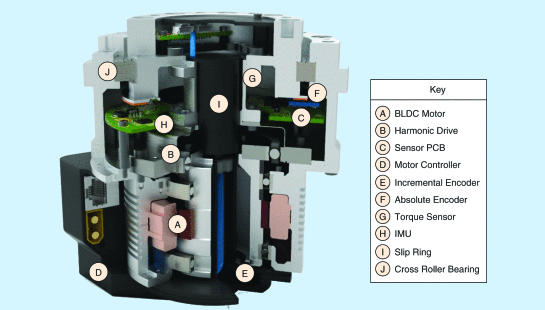
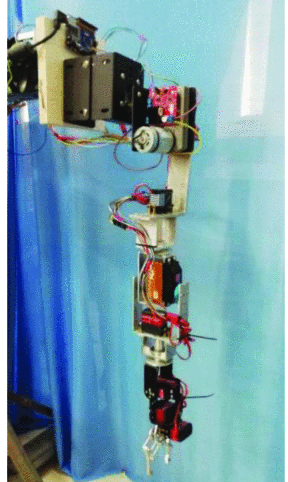
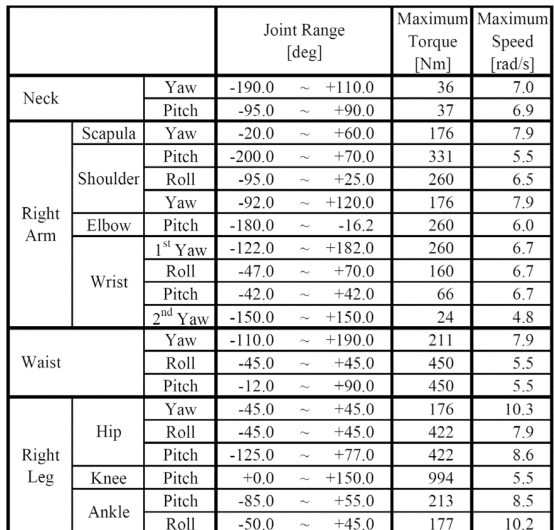
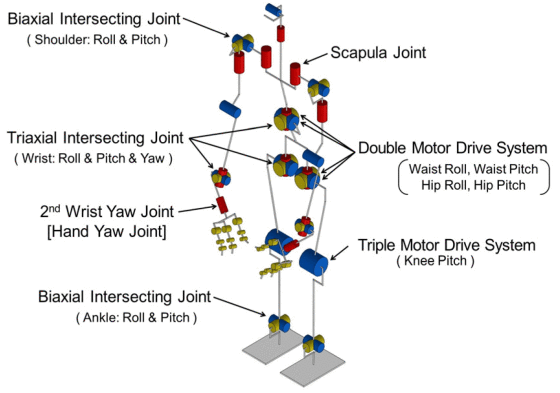
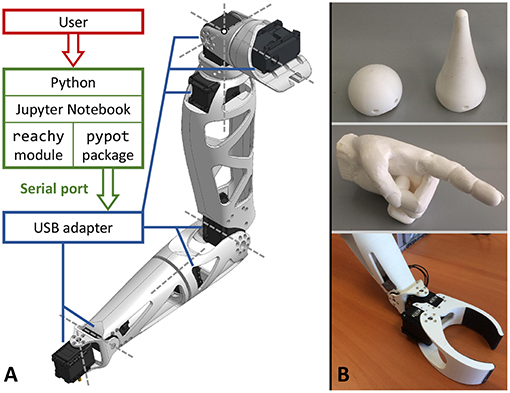
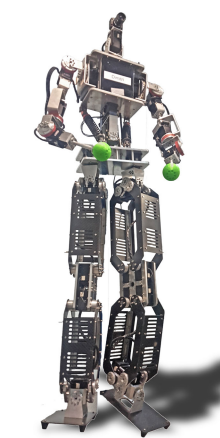
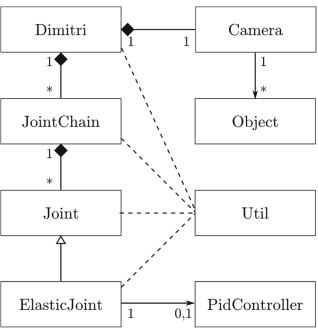
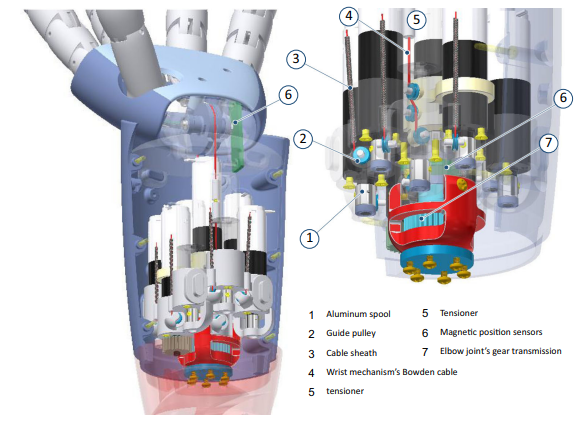
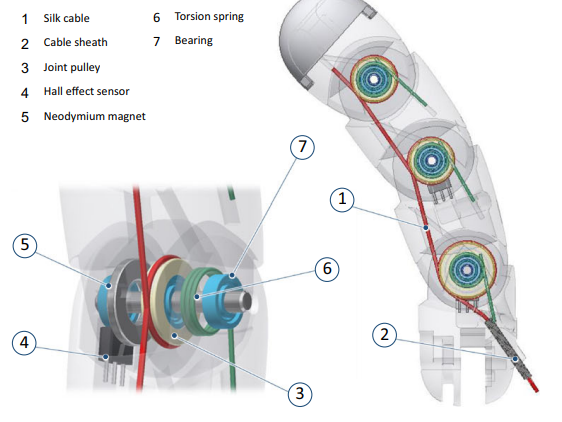
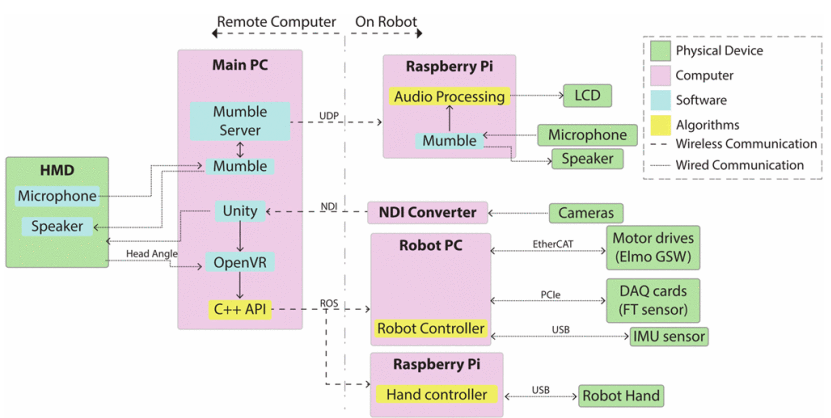
* Soft robotics
* Prosthetic interfacing
* Control gloves and similar
* Complex path planning
* Humanoid movements
* Human-robot interactions and cooperation
* Non-humanoid arms
* Generating human-like motions for aesthetic or cooperative reasons (naturalistic movements)
* Machine vision, audio sensing etc
* Walking etc
* Autonomous behaviour

Robots

<https://link.springer.com/chapter/10.1007/978-3-319-93870-7_13/tables/2> - table of <2018 robots

*These include the HRP (Humanoid Robotics Project) robots, used for aircraft assembly tasks [6] and construction work [7]; DLR’s Rollin’ Justin [8]; the fully torque-controlled TORO [9]; and robots such as the WALK-MAN [10] and E2-DR [11]. (Asfour et al., 2019)*

(Saeedvand et al., 2019) has another list

* (Asfour et al., 2019) full humanoid (ARMAR-6) 2019
  + Performance
    - *to the best of our knowledge, no other humanoid system currently exists that combines an arm reach of more than 1 m with a carrying capacity of more than 10 kg at full arm extension to go along with its limitless joints.*
    - No individual control of fingers, (only 1 or 2? Motors for fingers) but autonomous mechanical adjustment of torque
  + Mechanical
    - Exoskeleton design
    - *Hence, most of the structural hand parts are 3D printed from durable polyamide using selective laser sintering. All of the cables are hidden in the interior of the hand to avoid electric failures. The hand itself is covered by a protective work glove.*
  + Actuators
    - Fingers
      * *Dyneema tendons guided through polytetrafluoroethylene tubes for reduced friction.*
      * 3 joints per finger, 2 for thumb
      * 4 fingers and thumb operated by one motor, torque is distributed using
    - BLDC and brushed motors with harmonic drives
    - Specialy designed devices (pictured) in three sizes (T from 63-176Nm and 13-34RPM)
    - Has limitless joints for shoulder, uper arm and forearm, (seams unnecessary but maybe for screwing in screws etc
    - 
  + Sensors
    - Absolute encoders and incremental encoders combining for res 0.1 deg
    - Each of the main drive motors includes a strain gauge based torque sensor with a sensitivity of 0.04Nm. Enables torque control at 1kHz
    - *6d force torque sensor* after wrist joint
  + Electronics
    - 4 miniature PC’s and one GPU plus network pherepherals
  + Software
    - *All of our recent humanoid robots [2], [23] share the same cognitive and functional software architecture implemented in ArmarX [24], [41]. As a complex humanoid robot system, ArmarX is not only a robot middleware but also a complete functional cognitive architecture offering ready-to-use modules on multiple levels of abstraction.*
* **(Prabhu et al., 2021) robotic hand with micro servos and control glove**
  + Performance
    - 1 DOF wrist, plus a rotational DOF
    - 5 DOF fingers (one for each finger inc thumb)
    - 9grm SG90 servos for fingers
    - Looks like MG996r servos for wrist
  + Mechanical
    - PLA FDM printing
    - Nylon wires pulled by servo horns in palm
  + Actuators
    - Miniature hobby servos
  + Electronics
    - Arduino
    - 16 channel servo driver PCA 9685
    - No sensors
  + Control
    - Flex sensors on glove
* (Naoki Fukaya et al., 2013) Hand with finger force automatically controlled via mechanical means
* Yang et al. (2019) Arm with 8DOF (only 1 for fingers)
  + Performance
    - Looks nothing like an arm
    - 
    - Can hold a 300g bottle of water
    - About 1cm error when stacking cubes due to gears and part precision
    - *LWH-Arm 0.4 features full-size, lightweight, with proper load capacity and low-cost, i.e., it is 0.81 meters long, 3.5 kilograms weight, and cost less than $1000*
  + Mechanical
  + Actuators
    - Look like brushed servos
  + Electronics
    - Arduino (lower level) plus PC (higher level)
    - 16 way driver
    - Doesnt look like any sesnors
  + Control
    - Arduino only rotates servos to target angles based on targets given to it via the PC
    - Controlled via skeleton framework from xbox kinect
    - Used Open Dynamic Engine (write about this) *an industrial grade, open source, high performance engine that supports precise simulation of rigid body dynamics, and is widely used in research on robotics simulators, games, etc [9]*
    - Windows???
* (Kaneko et al., 2019) full humanoid robot
  + Performance
    - *aiming to realize the use of practical humanoid robots in place of humans within large-scale assembly industries such as construction sites, aircraft facilities, and shipyards.*
    - Can lift 11kg gypsum boards
    - 
    - 
  + Mechanical
    - Belt drive to oil lubricated harmonic gearing using electric acuators
    - 
    - Oil is better than greece here for faster harmonic drives (40-50%)
  + Actuators
    - Used multiple motors linked together with a belt drive to get better shape conformity where needed
    - Motors are cooled by air fans
  + Electronics
    - Sensing unclear
  + Control
    - 2 mintiruer PC’s controlling motors drivers with integrated torque feedback and the PC’s reading the sensors through usb (with a protocol converter)
* (Shut & Hollis, 2019) UR-5 style arms on an omni wheel robot
  + Performance
    - *The system includes a pair of 7-DoF arms. Each arm weighs 12.9 kg, with a reach of 0.815 m, and a maximum payload of 10 kg at full extension.*
    - This payload is limited to 6.8kg however as the fingers buckle here
    - *3- DoF shoulder, 1-DoF elbow, and 3-DoF wrist,*
  + Mechanical
    - Has ball detent torque limiter
  + Actuators
    - BLDC SENSO-Joint 100 and 75
    - Hands
      * Barrett Hand BH-280 [20]
    - *TQ-RoboDrive BLDC motor, Harmonic Drive, cross-roller ring bearing that decouples the input and output, incremental and absolute encoders, motor temperature sensor and output torque sensor in a compact lightweight package.*
  + Electronics
    - *The motor controller board includes the driver electronics, 6-axis IMU sensor, as well as the communication and sensor interfaces. The only connections out of each sensor-actuator-control unit are to the DC-bus for power supply and to the Ethernet bus for communication. Those connections are daisy-chained between units and the cables are routed through the actuator's hollow shaft. This allows for ±720° joint rotation, minimum cables and overall smaller arm volume.*
    - *Intel Core2 Duo @ 2.4GHz, running Ubuntu 14.04 Linux*
    - Ethernet to inner controller
    - Has off the shelf hand with sensing and stuff
  + Control
    - Torque controller where torque is a linear function of pos error, speed error and gravity compensation (minus currently measured torque)
      * With matrices and vectors
    - 10hz PI control loop to control current and hence torque
    - Some other PD controller to check for “safety limits”
    - Doesnt mention a middleware
* **(Mick et al., 2019) Pretty arm that is primarily a testbed**
  + 
  + Performance
    - free and open-source sharing of both software and hardware resources.
    - *Reachy's motors can sustain up to 10 min of continuous operation and are able to work for as long as a full day when tasked with short, out-of-charge movements*
    - *They provide a payload capacity of about 500 g at endpoint level, that the robot can handle for a few minutes.*
    - *Out of charge, Dynamixel motors can reach a maximum speed of 500°/s and a maximum acceleration of 10,000°/s2. When they operate in their nominal angular speed range, their performance allows the robot's joint to reach their goal positions with a delay from 50 to 100 ms.*
  + Mechanical
    - PLA, open for cooling and scr
  + Actuators
    - Robotis Dynamixel servomotors
      * *allow the individual tuning of an internal Proportional-Integral-Derivative (PID) controller, maximum torque and mechanical compliance.*
      * Also used by [Ha et al., 2011](https://www.frontiersin.org/articles/10.3389/fnbot.2019.00065/full#B24); [Ly et al., 2011](https://www.frontiersin.org/articles/10.3389/fnbot.2019.00065/full#B36); [Hild et al., 2012](https://www.frontiersin.org/articles/10.3389/fnbot.2019.00065/full#B26); [Schwarz et al., 2013](https://www.frontiersin.org/articles/10.3389/fnbot.2019.00065/full#B49); [Dawson et al., 2014](https://www.frontiersin.org/articles/10.3389/fnbot.2019.00065/full#B15))
  + Electronics
    - *Reachy's motors are connected with each other in a series using three-pin connectors and powered by a pair of 12 V × 5 A power supply units, for a total power of 120 W. At one end of the series, a USB adapter allows for plugging into a computer.*
  + Control
    - Pypot (Python obviously)
      * Goes well with the motors
      * *Pypot also includes features to operate a virtual robot within the simulator V-REP*
    - Open source, others in Simulink
* (Tatsch et al., 2017) Dimexel full humanoid basic
  + 
  + Performance
    - 2.5kg max payload
    - 31 DOF
    - 4 SEA’s in each arm
    - Dropped a brick on its hand and it was fine
    - “Cost effective”
    - SEAs slowed joint speed,but allowed torque sensing and robustness
      * *A noticeable downside is the propensity for oscillatory behaviour frequently observed on the shoulder roll joints. This oscillations were decreased by adjusting PID gains, with the disadvantage of decreasing the system’s dynamics.*
  + Mechanical
    - Routed plastic springs in SEA joints create robustness to impacts
  + Actuators
    - Dynamixel MX-106R and MX-64R
      * Up to 10Nm
    - SEA is a motor with torsional spring for measuring output torque
  + Electronics
    - *SEAs’ feedback circuit using a RS-485 bus running the Dynamixel protocol*
    - *An embedded NUC computer is used to handle communication with the actuators and SEAs at 1Mbps using a RS-485 to USB adapter*
      * NUC computer is a small form factor complete PC
    - 12.6V 2000W
  + Control
    - Uses PID control for actuators with spring feedback
    - 
* Sign language robot (Meghdari et al., 2018) RASA **(Robot Assistant for Social Aims).**
  + Performance
    - upper-body of 29 DOF
    - “Relatively low cost”
    - Only 1 DOF per finger for bending
      * Honestly from the pictures this makes it look pretty crap at doing sign language
    - Included finger abduction
    - Thumb was included as flexion and opposition (2 DOF)
    - Excluded lateral movement of wrist but kept 6 DOF for each arm
  + Mechanical
    - SLS Polyamide 3D printing
    - Steel shafts
    - Al for upper body
    - Bowden cables for hand movements
  + Actuators
    - For the hand movements the actuators are all in the forearm *Nine 1.2 and 2.5 Watt Maxon brushed DC motors coupled with planetary gearheads, with reduction ratios, ranging from 1:67 to 1:131, were selected to drive the respective joints.* Plus cable driven wrist
    - 
    - Upper joints MX64 and MX28 Dynamixel servos
  + Electronics
    - *Servo control of each finger is realized by the use of custom-designed sensor modules placed at the first two joints of robot’s index, middle, ring and little fingers, as well as two joints in each thumb.* Plus one more at the pinky for abduction
    - Hall effect sensors, easier to fit in small spaces with 3D printing, aligned vertically with diametrically magnetised magnet. Only works for less than 80 deg
    - No need for force sensors
    - 
    - 12 bit magnetic AS5162 sensors from Austria Microsystems for wrist and elbow joints
    - Noise is minimising by digitising sensor data on boards close to the sensors
    - Motor controllers for forearm fit in the forearm
      * based on two 32 bit Atmel’s SAMD21 microprocessors
      * *nine DRV8871 H-bridge drivers equipped with voltage, current, and temperature safety features as well as current regulation circuitry … up to o 4 A at 12 V*
    - Some sort of PC
* **(Sakhineti & Jayabalan, 2020) SHARLA Torso with wheels and arms for social interactions**
  + Performance
  + Mechanical
  + Actuators
    - Brushed hobby servos
  + Electronics
    - Arduino, computer, servos so no motor controller needed
    - No fingertip sensors
  + Control
    - Gui in Unity3D editor + usb joystick communicates with arduino via com port
    - Play and record functionality
* (Schwartz et al., 2022) **Full humanoid TOCABI**
  + Performance
  + Mechanical
  + Actuators
    - Harmonic drives that can be backdriven
    - Kind of unlcear
  + Electronics
    - 48V motors, computer and sensors on 24V, 12V sensors, 5V raspberry pi etc,
    - Higher voltage obviously better as lower current
    - *The typical running power draw of the computer system is 120 watts. TOCABI has 33 Elmo gold solo whistle motor drivers that consume 150 watts in their idle state for its Microcontroller Unit (MCU) and peripherals.*
    - Motor controllers: *Elmo gold twitter and ADVANCED Motion Controls Flexpro* are newer better options, but the cabling is finicky
    - No torque sensors at joints
    - *embedded PC with an i7-9700 intel CPU running Ubuntu 18.04 and the real-time operating system Xenomai-3.0.10*
    - 
    - Raspberry pi(s) for lower level control
  + Control
* (Hayosh et al., 2020) cheap laser cut torso
  + Actuators
    - Dimexel
  + Electronics
    - Raspberry pi
* Below here is new and need to add to electronics paragraphs
* Procter and Secco (2021) Medical teleoperation arm <https://pdfs.semanticscholar.org/a546/3d41128a6382eaf5ef9715244593f64c535d.pdf>
  + Performance

*After calibration and enabling closed loop control, the actuator was easily capable of overpowering my hand trying to halt rotation. The arm was then tested on its own. At a mass of 1420 g and a length of 165 cm, assuming that the mass is evenly distributed across the upper arm, the arm applies an average 13.72 N force at a distance of 82.5 mm, for a holding torque of 1.162 Nm. The purpose of this test was to see at what point the actuator could life the arm itself, but, the arm moved in a jittery fashion until around 3 A even if that maybe occurred because of the friction and inertia from the gears. To test the actuator more harshly, we tied a 2.5 kg weight at the end of the elbow joint. That is 24 N of force at 0.29 m, for 6.96 Nm of torque on top of the 1.162 Nm from the arm. We then incrementally increased the current until I found a level that could lift the weight to some degree. The unbalanced load made the situation cumbersome, but the number we reached was 8 A. Extrapolating 8 A being capable of 6.96 Nm of torque, the maximum 25 A of the motor should theoretically be capable of around 21.75 Nm of torque. This exceeds the maximum rated torque of the gearbox at 20 Nm. At 20 Nm however the arm should be able to hold 5 kg at the wrist, with the arm fully outstretched.*

* + Mechanical
  + Actuators
    - BLDC
  + Electronics
    - Open source motor driver platform named Odrive
    - The key driver of this project. Each ODrive supports 2 motors and 2 encoders in SPI daisy chain configuration. Each ODrive is in turn slaved to single Arduino Every microcontroller which distributes commands it receives from ROS on my desktop PC. Table 4 reports the main parameter of the driver.
    - No torque sensing
    - No fingertip sensors mentioned
  + Control
    - ROS
    - SPI interface
* (Aller et al., 2022) full humanoid <https://www.frontiersin.org/articles/10.3389/frobt.2022.898696/full>
  + Performance
  + Actuators
  + Electronics
  + Control
    - ROS
  + Article is only about the algorithm for standing up )-:
* Introduction 0.75

This literature review has been undertaken as part of a honours engineering project at the University of Waikato to develop a pair of humanoid robotic arms. The project’s aim is to produce a pair of arms that can play the piano as a demonstration of their dexterity and control accuracy. The project focuses on the mechanical, electromechanical and low level control of the arms, there are no plans to develop high level control of the arm for autonomous or semi autonomous tasks. This is left as a latter exercise so the piano playing will likely be done mostly via record and play functionality using a digital twin. Two literature reviews have been undertaken for this project, and the combined aim of the two is to develop an understanding of the latest advancements in areas relevant to humanoid robotic arms. The other review focuses mainly on the mechanical design and joint mechanisms while this review covers:

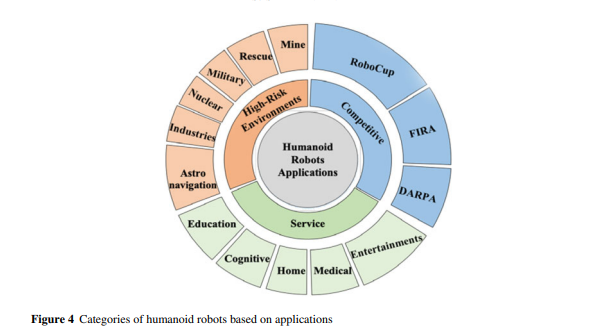
* Humanoid arm applications and introduction
* Actuators used in humanoid arms
* Joint sensors
* Tactile sensors
* Control electronics used in humanoid arms
* Robotic software environments
* Low level control software algorithms and methods

Conversely the following is out of scope for this review:

* Soft robotics. - only write about general to project here
* Control input methods such as prosthetic interfacing, control gloves etc.
* Complex path planning. - combining together maybe
* Human-robot interactions and cooperation.
* Non-humanoid arms. - more detail
* Generating human-like motions for aesthetic or cooperative reasons (naturalistic movements).
* Machine vision, audio sensing etc.
* Autonomous behaviour.

  + Context
    - *As examples drawn from some of the most advanced devices currently on the prosthesis market, Michelangelo (Ottobock) and i-limb quantum (Touch Bionics) hands include too many actuators for an amputee to operate them independently, and their control relies a lot on pre-programmed grip patterns. Even in the case of an able-bodied human, the gap between robotic devices' complexity and available command signals highlights the need for efficient and usable control interfaces and strategies. (Mick et al., 2019)*

(Saeedvand et al., 2019)



* + Focus and purpose of review
  + Table of contents or similar
* Overview of robotic arm technology (write this near the end) 1.5

Humanoid robotic arms have many uses, but the human arm is not the optimal topology for a robotic arm so they only have uses where the arm’s similarity to a human arm is useful. One of the most obvious is the applications in prosthetics (Aljalal et al., 2020), where the ultimate aim is to allow the user to have control of a robotic arm that replicates a human one to the same degree that humans have control over their own arms. Another key operation is teleoperation such as the medical teleoperation arm from Procter and Secco (2021) that aims to allow remote medical operations by skilled surgeons without needing them to travel (does it?). Similarly such teleoperation robots allow a skilled human to operate the robot remotely without having to travel (if there are few personnel with the appropriate skills for the task), which costs money and potentially downtime, and without potentially putting themselves in danger. There are also robots that have been developed to be sign language interpreters or teachers (Meghdari et al., 2018), where dexterity and humanoid looks are paramount, though these appear to currently lack sufficient abilities to be useful. A forth application is that for robots whose primary purpose is human interaction, humanoid robots can make the robots appear more friendly and personable An example of this is the human interaction robot from Sakhineti & Jayabalan (2020) which is designed to provide companionship and assistance to the elderly (including those with dementia) and those with mental disabilities. Finally, humanoid robots allow for the straightforward replacement of humans in tasks and environments that have already been designed for humans. For example in low skill factory jobs currently done by humans that require high flexibility (Ali Ahmad Malik et al., 2024). In many of these situations a non-humanoid robot would be more effective but a humanoid robot has the advantage that its programming is more intuitive and it does not need to be custom designed to undertake tasks that were previously done by humans.

Extensive research has been undertaken on humanoid arms in recent years and they can be evaluated by various metrics. The first is the degrees of freedom in the joints, and how well the joints correspond to the human joints. Of the well funded top of the line robots, nearly all have a 6 or 7 degree arm (including the wrist) (Stasse & Flayols, 2018), which very closely matches the joints in the human arm. However almost none come close to the number of actuated (not including flexible joints that can be controlled) degrees of freedom (DOF) in the human hand, generally listed as 21 actuated degrees of freedom (Breen et abe moved l., 2003)(Rahman & Al-Jumaily, 2013), which includes 3 extension and flexion motions in the fingers, an abduction/adduction motion in the fingers and 5 DOF in the thumb. The most dexterous robots towards this end are robots such as the Walkman robot from the Istituto Italiano Di Tecnologia (IIT) which has 19 DOF per hand (Stasse & Flayols, 2018) and the ShadowHand which has 18 actuated DOF per hand (Shadowrobot, n.d.)(not counting the DOF in the wrist). There are also non-actuated DOF of which the shadow hand has an additional 4. These are clearly a very similar numbers to the DOF to the human hand however hands like the ShadowHand are considerably more bulky than a human forearm and don't have anywhere near the same strength as a human hand: it can only hold a maximum of 5kg (Shadowrobot, n.d.) and the Walkman robot can only hold a maximum payload of 10kg (Davis, 2018). There is considerable uncertainty in these numbers as well as it is not specified in what grip type the robots can hold these loads. This highlights a technological gap as there are no robotic arms in existence that have both the same dexterity and strength as the human arm, as 10kg is far below what a strong adult human could lift and manipulate. Generally there is a trade off between gripper strength and gripper dexterity, so most robotic arms have considerably under actuated hands, which are also cheaper and less complex (Stasse & Flayols, 2018)(Asfour et al., 2019)(Prabhu et al., 2021)(Naoki Fukaya et al., 2013)(Kaneko et al., 2019)(Tatsch et al., 2017). Many of these arms also hold similar payloads like dual arms from Shut & Hollis (2019) which can hold 6.8kg using the fingers of a hand, but up to 10kg if the weight is attached to the end effector. Another robot with similar weight capabilities is the full humanoid robot from Kaneko et al. (2019) which can lift 11kg gypsum boards. The arms with higher DOF at low price points are often sign language robots such as from Meghdari et al. (2018) which has 9 actuated DOF in each hand, in these types of robots there is very little torque in the fingers as they are not designed to grip objects. Most robots at a low price point have actuated DOF in the hand less than 5. This does highlight another gap in the literature where there are few hands with a moderate to high degree of freedom and moderate strength at a low price point, and is something we hope to address with our robotic arm.

Another metric by which robotic arms can be evaluated is by how accurately they can move and manipulate objects. Unfortunately a broad overview of data on this topic for humanoid robots is missing in the literature and data is difficult to come by as most papers on individual robots quantify their accuracy in unique ways that are not easily comparable or entirely neglect to quantify the robot’s accuracy. A review on the accuracy of hydraulic robots has been done by Mattila et al. (2017) in which the most accurate robot was said to have an accuracy of 1.5mm at the end effector, however the reference listed did not provide any support for this. Other hydraulic robot end effector accuracies from this review were listed as between 5.2-120mm. The low cost hobby brushed servo arm from Yang et al. (2019) listed an accuracy of +/-10mm when stacking blocks which was attributed to gears and part precision, which is relatively accurate considering that the actuators used do not appear to be robotics grade. Industrial robotic arms, while not the focus of this review, provide a good indication of achievable accuracy where data for humanoid arms is lacking, and accuracy for these can be as high as +/-0.1mm for some high quality electric actuator arms (Lattanzi et al., 2020). +/-0.1mm is clearly sufficient to replicate the accuracy of a human arm, but most humanoid arms are unlikely to be this accurate due to space constraints limiting the size of the actuators and structure of the robot. Hence it is unclear how common high accuracy is in humanoid robots.

Uses 0.5

* + - Picking up objects (Kamil Khusnutdinov et al., 2019) (simulation only, but based on real robot)
    - Sign language robot (Meghdari et al., 2018)
    - Automation where high flexibility is needed and human centric environments (Ali Ahmad Malik et al., 2024)
    - Teleoperation
    - Prosthetics another reference?
    - companionship
  + Effectiveness - how good are they currently - 1 page to write
    - DOF
    - Strength
    - Accuracy 0.3
* Use of different actuators, their control characteristics and required electronics 3
* Critiquing sources

Arguably the most important electrical components in a robot are the actuators. They determine the strength of the robot, the size, the majority of the robot’s weight, the positional accuracy of the joints, and the safe duty cycle.

#### Motor requirements

The key requirements are generally a high power to volume ratio and low speed-high torque operation (Stasse & Flayols, 2018). For robots that are mobile, power to weight ratio is also important (Shut & Hollis, 2019) as lower weights allow for a faster movement and lighter, cheaper robots. Another example where weight is important is of course prosthetics. Even for non-mobile robotics such as this project, lower weights are desired as it increases safety when interacting with humans and increases lifting capacity. Backdrivability (which is not possible for very high gear ratio gearboxes or worm drives for example) is also desired characteristic (Stasse & Flayols, 2018): it allows for the robot to be robust to collisions, to interact with humans in a more natural way and for better control of torque when torque sensing on the motor side. It is not however a requirement as collision robustness can be achieved via torque limiting mechanisms or via flexible couplings (Tatsch et al., 2017) and torque can be managed by sensing torque on the output side (where it is unaffected by the inability to backdrive). They should be robust (Shut & Hollis, 2019) and for our purposes relatively low cost.

Leaning towards the control requirements for actuators, the actuators must be capable of accurate position control (Shut & Hollis, 2019) in a robotic arm due to the compounding effect of having so many DOF, which can either be achieved through embedded sensors or through sensor feedback from the endpoint of the actuator. Ideally they should also have some sort of torque control as this allows for differing torques for grasping different objects and for limiting torque when working with humans to increase safety. This review will focus mainly on these control characteristics across the spectrum of different actuators available.

#### Electric actuators

Electric actuators appear to be the most popular choice for humanoid arms. This is due to their simplicity, low cost, high degree of available options and ease of control. They are most commonly DC motors, either brushed or brushless, but AC motors are also used (Stasse & Flayols, 2018) as they can offer higher performances despite the need to transform DC battery voltages for mobile robots. DC motors can be either standalone or servo versions (brushed or brushless) that have integrated position feedback and sometimes torque feedback or control. Finally stepper motors are also used, which are fundamentally the same concept as brushless DC motors but with more poles, and solenoids are another option that is rarely used in humanoid robotics due to its limited positional control. Talk about controllers and power for each motor

Brushed DC motors are used in some robots such as Meghdari et al (2018), a sign language robot and Asfour et al (2019). They are usually purchased as a gearbox-motor combination, where the gearbox is usually planetary. They offer the advantages of low costs and relatively straightforward combined speed-torque control via PWM or voltage adjustment. To get position control they require extra rotational displacement sensors and a position - motor power feedback loop which is commonly PID (Meghdari et al., 2018) which is their main disadvantage. They offer medium power to weight and power to volume ratios, mainly due to space not being wasted with inefficient sensor placement like it is in servos.

Brushless DC (BLDC) motors as standalone units offer the advantage of the highest power to weight and power to volume ratio of any electric actuator however the trade off is control difficulties (Procter & Secco, 2021) and higher costs. They are used in robots such as the full humanoid from Asfour et al (2019), where they allow the robot to have very high power joints. They are also used in the medical teleoperation robot from Procter and Secco (2021), which allows for the use of high powered hip joints powered by drone motors with added encoders. In this robot they were controlled using an open source motor driver called Odrive. No overheating issues were mentioned in the paper but the use of drone motors under high loads tends to incur heating issues as they are designed for low torque, short (7-30 min) duty cycles and to have large amounts of cooling from the attached propeller forcing air through the motor (Paul, personal communication, March 5, 2024). The lack of heating issues was likely because the motors were only tested for short periods of time as the arm was not fully operational. BLDC motors come in either inrunner or outrunner versions, where outrunners have the rotor on the outside and vice versa for inrunners. Generally for robotics applications outrunners are used as they offer higher torques and lower speeds (Procter & Secco, 2021). They are also sold either with or without embedded hall effect sensors. Control of the motors at stall or low speeds or stall is realistically possible only with sensored versions (though of course sensors can be added), as the back EMF effect needed in sensorless versions to determine which poles to apply a voltage to only works when the motor is spinning. These sensored versions generally cost more and are not available for low power ratings which limits the applications of BLDC motors to the larger joints in a robotic arm. In the full humanoid robot from Asfour et al (2019) they are controlled using feedback from an absolute encoder, an incremental encoder and a torque sensor, which allows for very precise control, but at a high cost. It is likely the second incremental encoder is just used here to increase accuracy as in theory a single absolute encoder would be sufficient for positioning.

Likely the most common actuator for robotic arms is instead the BLDC servo motor. These are generally robotics grade versions (though hobbyist, RC vehicle grade servos are also used) and the most common are those from Dynamixel used in robotic arms from Hayosh et al (2020), Meghdari et al (2018), Tatsch et al (2017), and Mick et al (2019). These motors have high power to weight and power to volume ratios, have position control with integrated PID controllers, have torque controller or torque sensing, and continuous rotation. These are typical characteristics for motors of this type, though not all will have torque control or torque sensing. The dynamixel motors in particular also can be chained together to minimise cabling and come with extensive robotics software to make them easy to use (Mick et al., 2019). These Dynamixel motors use serial communication and in the literature examples are controlled via a Raspberry Pi or a miniature computer. Of course other similar BLDC servos are also used for robotic arms for example in the arms developed by Shut & Hollis (2019), which make use of large diameter cylindrical servos as the function of their arm is not dependent on the arm appearing humanoid.

Brushed DC servo motors are a cheaper alternative to BLDC servos that offer many of the same advantages and multiple humanoid robotics projects built on lower budgets use these motors such as a robotic arm from Yang et al. (2019) and a robotic hand from Prabhu et al. (2021). They also come in robotic grade and hobbyist grade versions, however robotics grade servos appear to be more likely to be BLDC. They have a lower torque to volume and torque to weight ratio than their BLDC counterparts. And while they do have PID controlled (?) position control, the hobbyist grade versions used in the mentioned robots lack any sort of torque control, torque sensing or ability for the user to customise the position control algorithm. They don't usually come in versions where the wiring can be chained together either.

Other DC electric actuators include solenoids and stepper motors. Solenoids find little use in robotic arms as generally they don't have any position control. Stepper motors are not commonly used either, most likely as they are not precise enough to replace high quality servos, and not powerful enough to replace standalone DC motors.

AC motors can deliver high amounts of power, however the trade off is more complex control (Stasse & Flayols, 2018). This is because their torque rapidly drops off at lower speeds and they tend to have low torque on startup. Some robots do use them however, such as the TORO humanoid robot, which uses torque sensors to make the control more feasible (Stasse & Flayols, 2018).

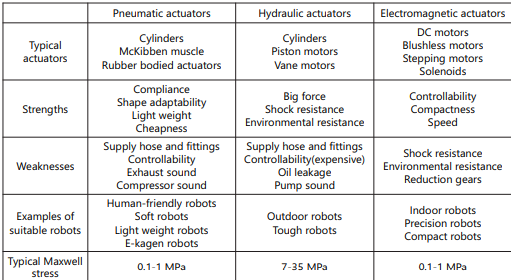
Hydraulics are another alternative to electric actuators and are used in many high performance robots such as the Boston Dynamics Atlas robot (Stasse & Flayols, 2018). They offer better power and torque per kg and per cm3 than electric motors (Stasse & Flayols, 2018) (Junget al., 2018). They also offer good control of output force through the use of variable pressure (electric) control valves (Stasse & Flayols, 2018). Unfortunately they also come with significant downsides. The most important of these is their high costs (Saeedvand et al., 2019), their tendency towards leakage problems (Kaneko et al., 2019) and greater difficulty in position control (Mattila et al., 2017). Mattila et al (2017) postulates that position control can be difficult as in many examples of hydraulic robots the position of a hydraulic actuator is intimately and non-linearly linked to the force on the actuator, which induces difficulties especially when the actuator is interacting with objects (and the force is being changed). Error at the end effector in a range of 6 research arms from 2002-2015 was found to be mostly between 1.5-27mm with one arm having an error of 120mm, though vastly different working velocities are used for many of the arms which would have an impact on the accuracy (Mattila et al., 2017). So while the control may be difficult (indeed many of the arms use nonlinear model based control systems and there are many different controller methods used) high accuracy is clearly achievable, if at a higher cost and complexity. Other downsides to hydraulic arm systems include the large size of hydraulic pumps which is a limitation for mobile robots (Stasse & Flayols, 2018), the loud sound of hydraulic pumps (Stasse & Flayols, 2018)(Suzumori, 2020) and the low energy efficiency (Mattila et al., 2017). Energy efficiency is important to reduce operating costs, environmental impact and for mobile robots: battery life. This efficiency is a key limitation of hydraulics not considered by Stasse & Flayos (2018) in their overview, however they do consider how pump size limitations for mobile robots can be overcome by well designed pumps such as from Alfayad et al. (2011).

A method for improving hydraulics is instead to use electric hydraulic actuators (EHAs). These actuators use servo motors to push fluid to drive a hydraulic actuator. Clearly this increases costs and complexity even more as it has twice as many actuators per joint, but it does offer a few advantages for high accuracy robotic arms.

Write some more on EHA here.

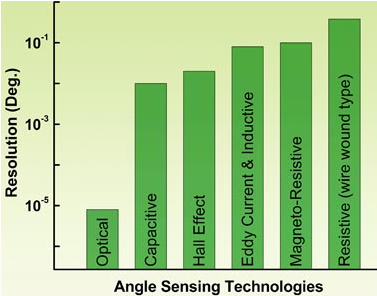
Similar to hydraulics are pneumatic actuators. These offer advantage over hydraulics of less concern over leakage but they are not as easy to control position (Hashimoto, 2020) and torque (Stasse & Flayols, 2018). They also generally offer faster actuation than either hydraulic or electric actuators (Mori et al., 2018). One of the most common pneumatic actuators for robotic arms is the Mac Kibben muscle (Tondu, 2012), which has a form factor reminiscent of human muscles and thereby can allow for more humanoid movements and actuator placements. Tondu (2012) further highlights the difficulties associated with accurate control of this “muscle” in their paper, due to such issues as the non-linear stress-strain relationship of the elastomer used in the muscle, and the muscle hysteresis. Pneumatic actuators are also the key actuators for soft robotics (Su et al., 2022) which is an important emerging field and especially relevant to humanoid arms that will interact with humans. For these soft robots powered by pneumatics either muscle type actuators connected to cables are used or soft fillable voids are used. These soft robots offer superior resistance to damage, and safety over traditional solid robots. Robots such as the badminton robotic arm from Mori et al. (2018) also make use of pneumatics as this allows for the fast actuation needed to play badminton. This team also managed relative accurate control on the order of 10-40mm using fast swings using feedforward control.

Including motor drivers

* + Requirements
    - *high ratio between power and mass, the capability to produce high torques at low speed, a relatively small size and back drivability (Stasse & Flayols, 2018)*
    - Weight is important for applications on mobile platforms (Shut & Hollis, 2019)
    - accuracy, speed and robustness in mechanical terms, as well as embedded sensors monitoring angular speed and position.
    - Low cost
    - Torque control
  + Brushed DC
  + AC motors
    - Higher performance? (Stasse & Flayols, 2018) more complex control
    - May have refrigerant liquid surrounding it (above,S-One from Schaft)
  + Brushless DC
  + Brushless DC drone motors
  + Servo
    - *Common successfully used servomotors in humanoid robots are the Dynamixel motor series1. (Saeedvand et al., 2019)*
    - Hobby vs robotic grade
    - Brushed
    - Brushless
  + Stepper
  + Solenoids
    - Standard
    - Positional control (custom?)
  + Pneumatics
    - *The most common pneumatic actuator is the Mac Kibben muscle [62]. It consists in an air chamber inside a highly resistant fabric sheath crimped at both extremities of the chamber. When this chamber inflates, the highly resistant sheath is in contact, and is contracting the muscle. This is producing a traction force related to the air pressure put in the air chamber. It is possible to use models developed in the pneumatic industry to model the relationship between the pressure and the generated force. However friction resulting from the interaction of the sheath and the air chamber is introduces non linear phenomenon which are making the control of pneumatic actuator difficult.* (Stasse & Flayols, 2018)
    - *The selection of actuators is most important in realizing such a robot. We employed pneumatic actuators. Among other commonly used actuators, an electromagnetic actuator has a quick response and high controllability. However, its speed is strictly limited by its back electromotive force and its power-to-weight ratio is low because of its heavy coil and magnet. A hydraulic actuator has a high power-to-weight ratio for an industrial machine, but it needs other heavy apparatus such as pumps and motors. Moreover, its speed is limited because of high oil viscosity. Meanwhile, a pneumatic actuator has a much higher power-to-weight ratio than other actuators on the whole, and can realize high-speed motion. Additionally, a slow response can be a serious problem for the feedback control, but we will propose a feedforward control method that is suitable for the high-speed motion of a pneumatic robot. (Mori et al., 2018)*
  + Hydraulics
    - 
    - It exits several humanoid robots using hydraulic systems such as DB and CB developed by SARCOS [9]. The most well known is the ATLAS robot from Boston Dynamics. (Stasse & Flayols, 2018)
    - Good
      * Power (Stasse & Flayols, 2018) *However, hydraulic actuators can make higher torque than an electric motor with the same size (Junget al., 2018).*
      * Force control (Stasse & Flayols, 2018)
    - Bad
      * Autonoumy (large pumps) (Stasse & Flayols, 2018) (not a big deal for us)
      * Leakage problems (Kaneko et al., 2019) (if they avoided it with their huge budgets we should definitely avoid it)
      * Pump sound (Stasse & Flayols, 2018)
      * Higher cost (Saeedvand et al., 2019)
      * Not energy efficient (Mattila et al., 2017)
      * *dynamic behavior of hydraulic systems is highly nonlinear, making their control, especially in constrained motions, a truly challenging task and (Mattila et al., 2017)*
    - electro-hydrostatic actuators
      * it is probable that such technology is used for the ASIMO hands presented in 2011. Similar works have been realized for the hands of the humanoid robots ARMAR (using air) [[35](https://link.springer.com/chapter/10.1007/978-3-319-93870-7_13#ref-CR35)]. More recently a reversible EHA including a torque sensor has been proposed at Tokyo University [[29](https://link.springer.com/chapter/10.1007/978-3-319-93870-7_13#ref-CR29)]. (Stasse & Flayols, 2018)
  + Cooling requirments
  + Pneumatic "balloons"
  + Cable driven actuators
    - Cable screw
* Sensing methods 1

Another key part of a robotic arm are the sensors implemented. These can be broadly broken into two categories: joint sensors and environmental sensors. Joint sensors are used on each joint and include sensors for position (and thereby speed and acceleration) and sensors for torque. Environmental sensors are widely varying but include sensors such as cameras, microphones, tactile sensors and time of flight cameras or scanners. This review will only focus on joint sensors and tactile sensors as these are the only sensors that will be implemented on the robotic arm the author plans to build.

Joint sensors for sensing position (rotational displacement) can be either standalone components added in to allow for position control of standalone motors, or they can come preinstalled in servo motors. It is absolutely key to a useful robotic hand as small errors in the position of one joint compound to large errors in the end effector position. They can also allow for damage detection (Stasse & Flayols, 2018). These servo motors will then have a basic controller inside them that will allow for (at a minimum) position control through the sensor feedback. Clearly this is a key advantage of using servo motors over other types, and their accuracy will depend on the servo selected. Regardless of whether they come preinstalled on a servo or not, angle sensors generally come in two forms: potentiometer based or encoder (hall effect or magneto resistive) based. These are not the only possibilities however and another commercially available alternative are inductive sensors. Capacitive and optical angle sensors have also been developed to a high accuracy (Kumar et al., 2021) (see figure graph) but their commercial availability appears to be severely limited. Potentiometer based sensors rely on the variable resistance of a sliding connector that rotates to different positions along a wire, whereas encoders rely on the changing magnetic field of one or multiple magnets that rotate. Hall effect based encoders are the most common angle sensor to apply to a BLDC motor to sense its angle (Kumar et al., 2021) and they come in either absolute or incremental versions (these can only sense the change in angle from the starting angle). Most robots use servo motors which avoids the need for additional sensors, but the custom motor-sensor combinations used in the ARMAR-6 and the HRP-5P full humanoid robots both use an absolute and an incremental encoder on most joints (Asfour et al., 2019)(Kaneko et al., 2019). Another example of encoders in use is on the Sign language robot from Meghdari et al. (2018), in order to maximise finger DOF in the fingers brushed DC motors have been used instead of servos (as servos are larger) and the joint angle sensing on the fingers has been achieved by using a hall effect sensor coupled with a diametrically magnetised magnet. This is essentially a custom encoder and has been made in this way due to the minimal space on the fingers for adding sensors. A similar design using 17 tiny hall effect sensors and ring magnets (figure coin) is employed for sensing the angles of the phalangeal joints in the highly dexterous iCub robot (Schmitz et al., 2010). Using sensors on the joints, which are actuated by cables in both robots, increases accuracy over rotation sensors on the motors used to pull the cables as joint sensors are not affected by cable flex and slack. On-axis absolute encoders are also used in the custom actuators on the medical teleoperation arm from Procter and Secco (2021). In terms of servo motors, potentiometer sensors are commonly used for hobby brushed servos (Adafruit, n.d.) while BLDC servos are more likely to use absolute encoders, for example on the Dynamaxil range of BLDC servos (Dynamixel, n.d.).

(Kumar et al., 2021)

(Schmitz et al., 2010)

Torque sensing is another important aspect of joint sensors as it allows for the detection of objects and, if coupled with torque control, allows for the robot to apply appropriate torque for interacting with objects. Hobbyist servo motors do not typically have torque sensing, however BLDC servos mostly have torque sensing in built, though they do not necessarily have variable torque output.Torque can be sensed based on a strain gauge such as the custom design used in the ARMAR-6 humanoid (Asfour et al., 2019), and in the BLDC servos commonly used in many robots (Shut & Hollis, 2019). Torsional springs with angular displacement sensors can also be employed as in some of the actuators on the full humanoid from Tatsch et al. (2017). It is not a requirement to have censored torque sensing however and many complex robots such as the humanoid TOCABI do not use them (Schwartz et al., 2022). If specific sensors are not employed then torque can also be estimated based on sensing the current going to the motors (this is less accurate due to non-linear rotational friction, and changing temperature response).

Tactile sensors are an area of ongoing research, especially in regards to large area sensors that mimic the capabilities of human skin (Roberts et al., 2021)(Pagoli et al., 2022). While tactile sensors are potentially useful on the external surface of any part of a humanoid robot, the most important locations to consider are the hands, specifically on the top pads of the fingers and thumb and on the top of the palm (Weiner et al., 2019). They are important as they allow for the force being applied to objects the robot is interacting with to be sensed as well as its compliance to applied force, and more complex sensors even allow the robot to sense when objects are slipping out from the fingers or to sense the temperature and texture of the object. Some sensors can even sense vibrations. Due to the size and form factor requirements they need to be thin and flexible (Hammock et al., 2013), and they also need to be soft to allow objects to be easily grasped (Roberts et al., 2021).

Types of sensors that have been used in or developed for robotic gripper applications include:

* Resistive
* Capacitive
* Magnetic
* Barometric
* Optical
* Accelerometers (used for slip detection)
* Distance sensors
* Temperature sensors

Of these capacitive touch and piezoelectric resistive type sensors are the most commonly used. (Stasse & Flayols, 2018) Capacitive sensors work by the compression of force applied to the finger bringing one electrode (usually flexible material in this case) of a capacitor closer to the other, thereby changing the capacitor’s capacitance (Roberts et al., 2021). They can also sense directional sliding forces (shear forces) due to the angle of the capacitive electrode changing (Hammock et al., 2013). Such sensors are used in the I-Cub hand (Schmitz et al., 2010) and are implemented using 108 pads on flexible PCBs and then covering these pads with first a silicone foam dielectric, then with a grounded conductive layer (silicone or lycra). In this case however the sensors were not designed to detect shear forces and their accuracy in detecting force was not quantified, only their ability to detect contact. Sensor communication was done over I2C with chips for each group of sensors to minimise wiring. One type of resistive sensor is the piezoelectric type consisting of piezoelectric materials with overlapping electrodes. This type has been demonstrated for haptic feedback robots (Roberts et al., 2021). Another type in the literature is the liquid conductor type that has a liquid metal or ionic conductor contained within channels in a polymer. As pressure is applied to the polymer the channels’ cross sectional area decreases, which increases the resistance through the liquid conductor (Roberts et al., 2021).

Magnetic sensors have either an elastomer embedded with magnetic particles on top of a hall effect sensor (Roberts et al., 2021) or a magnet on top of a flexible material on top of a hall effect sensor (Weiner et al., 2019). In the former case pressure is interpreted using machine learning algorithms while in the later case the interpretation is more straightforward but the sensor is presumably thicker. The former case also allows for the measurement of force and location of that force, down to sub millimetre accuracy (Hellebrekers et al., 2019). The latter can be used to measure both normal and shear forces through measuring field orientation as well as stress (Weiner et al., 2019). Another sensor type that is popular is a barometric type that measures fluid pressure from a fluid sealed in an elastomer (Roberts et al., 2021). This type is used in some commercial sensors such as the RightHand Robotics Inc’s TakkTile sensor and the BioTac sensor from SynTouch (Roberts et al., 2021), and the NPA201 from Amphenol (Weiner et al., 2019). Both magnetic and barometric sensors were used on the ARMAR-6 robot by embedding 2 into the distal phalanges and 1 into the middle phalanges of their robot and covering them with moulded silicone (Weiner et al., 2019). They partly used so many to test the effectiveness of the sensors but also because multimethod sensing showed advantages. On the same fingers they also implemented an accelerometer, polled at a high frequency of 1.6KHz, and using the short bursts of detected acceleration they were able to successfully detect when an object was slipping from the robot’s grasp. They also implemented time of flight distance sensors on the fingers which allowed them to conform the grip to the object better by sensing the distance from the base of the middle phalanx to the object. Capacitive sensors can also be used in this way to detect distances up to 60mm if designed well (Gao et al., 2020). Finally, in the ARMAR-6 each of the sensors had an additional temperature sensor to allow for calibration. Obviously all these sensors would have come at significant cost and complexity so there are many projects that would not justify such an array.

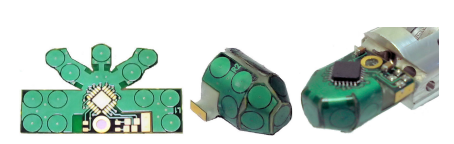
The last of these sensors are the optical sensors. These work by having a constant light shine against a deformable reflective surface and the light bouncing off that surface being measured. When force is applied the surface deforms and the light coming into the light sensors changes, a machine learning algorithm can then be used to detect deformation direction and distance (corresponding to force). They are still very much in the research stage (Cirillo et al., 2021) but some robots like the full humanoid NASA Robonaut 2 use them in combination with other sensors (Roberts et al., 2021).

In general not many of the humanoid arms in the literature integrate fingertip sensors, this is likely due to the focus of the research being on one aspect or the other. Nevertheless this does highlight a gap in terms of the testing of the sensors.

* + Positional sensors -1
    - Encoders
      * Motor end and joint end sensing allows for damage detection
      * Encoders are an essential part to evaluate the current robot state. A very good precision is necessary, specifically for the legs, in order to prevent impact during foot landing. Typically the theoretical articular precision of a robot such as HRP-2 is in the order of 1 / 1000 of degrees. (Stasse & Flayols, 2018)
      * Could add more about dust and vibration resistance if needed
  + Torque sensors
  + Pressure sensors - 1
    - Requirements
      * *soft, thin, flexible, stretchable, and lightweight (Roberts et al., 2021)*
      * Needed most in human-machine interactions, wearables and prosthetics, remote presence devices (Roberts et al., 2021)
      * Write about what sensing a human has?
      * Sensor placement (Weiner et al., 2019)
        + Finger pad is the most important, especially of the index finger
        + Mostly tested on round objects though of a certain size
    - Capacitive touch and piezo electric most common for detecting contact (Stasse & Flayols, 2018)
    - *with piezo-electric sensor, it is also possible to add a cover which is transmitting current and then act as a force sensor.*
    - *Using Micro-Electro-Mechanical Systems (MEMS) gives the possibility to measure force orientation, shear forces, vibrations and their duration. Finally adding thermal sensors provides temperature. One of the major blocking point with skin is the necessity to have efficient communication buses due to the large number of sensors needed to cover the surface of the robot.* (Stasse & Flayols, 2018)
    - Soft “skins”
      * Resistive (Roberts et al., 2021)
        + Liquid metal or ionic fluid in channels, CSA and hence resistivity of channels changes under pressure
        + Piezo resistive materials and overlapping electrodes

Piezoresistive inks are popular

Demonstrated for haptic feedback

* + - * + *Other methods include the use of foams [44,45,46], piezoelectric materials [47, 48], conductive hydrogel microspheres [49], and carbon nanotubes [50]. For example, a piezoresistive tactile sensor based on a hierarchical pressure-peak effect is described in [51]. With this approach, a wide detection range and a high sensitivity are achieved for detecting different pressure stimuli like foot pressure, respiration, and pulse and finger heart rate.*
        + *unreliable static sensing properties.* (Hammock et al., 2013)
      * Capacitive
        + *are typically composed of measuring the change in capacitance between two overlapping electrodes that are separated by a dielectric elastomer* (Roberts et al., 2021)
        + Can measure normal and shear forces (thorugh change in distance and area between the 2 “plates” (Hammock et al., 2013)
        + 
        + Used in (Schmitz et al., 2010)
        + *flexible printed*
        + *circuit board (PCB), which has 12 round pads, one for each*
        + *sensitive element, which are connected to a capacitive to*
        + *digital converter chip (AD7147 from Analog Devices), which*
        + *can collect and send 12 measurements of capacitance over*
        + *an I2C serial bus.*
        + *Above the flexible PCBs in the palm and the fingertips is a roughly 2mm thick layer of soft silicone foam (Soma Foama 15 from Smooth-On). It acts as a dielectric for the capacitive pressure sensor and makes the sensor compliant. On top of the foam there is a second conductive layer: electrically conductive Lycra for the palms, electrically conductive silicone for the fingertips (see Fig. 9). This layer is connected to ground and enables the sensor to respond to objects irrespective of their electrical properties*
      * Magnetic
        + Particles in a elastomer matrix that change the internal magnetic field under pressure (Roberts et al., 2021)
        + Relies on machine learning
        + *Tomo et al. [9,10] use commercially available digital three-axis Hall effect sensors together with a magnet embedded into flexible material above the sensor to measure normal and shear forces through the displacement of the magnet. These modular sensors were integrated into the Allegro [11] and the iCub hands and fingers [12,13]. (Weiner et al., 2019)*
        + *The shear force sensors (MLX90393, Melexis, Ypres, Belgium) are based on work presented by Tomo et al. [9,10]. They can be used to estimate both normal and shear forces and offer a larger measurement range than the normal force sensors.* Used in ARMAR-6
        + *Using a quadratic discrimination analysis, one study was able to distinguish between 25 grid locations in a 15-mm2 area with a > 98% accuracy*
        + *Recent studies have demonstrated the use of magnetic tactile skins for both localization and force feedback in robot grasping tasks [73]*
      * Barometric
        + *Long history in robotic manipulation* (Roberts et al., 2021)
        + *This approach is used in commercial pressure sensors like the TakkTile sensor from RightHand Robotics, Inc., and the BioTac sensor from SynTouch*
        + Fluidic medium sealed in an elastomer, the pressure of the fluid is then sensed
        + Used on ARMAR-6 as normal force sensors (NPA201, Amphenol, Wallingford, USA) (Weiner et al., 2019)
      * Optical
        + Flexible film with reflective surface coupled with camera
        + *combined photonic and barometric sensing incorporated into a sticker-like flexible circuit (adapted from Ref. [80] with permission, copyright IEEE, 2020). d The circuit is mounted to the fingertips of the NASA Robonaut 2 humanoid robot* (Roberts et al., 2021)
        + One from <https://www.mdpi.com/1424-8220/21/5/1915#B12-sensors-21-01915>
      * Accelerometers
        + Used to detect slippage through vibrations
        + ARMAR 6 (BMA456, Bosch Sensortec, Reutlingen, Germany) with a sample rate of 1.6kHz (Weiner et al., 2019) all sensors cast in silicone rubber
      * Distance sensors
        + *To gain additional information even before contact is made with an object, we implemented a distance sensor into the finger, shown in Figure 6. The sensor is a state of the art time of flight (ToF) device that is able to measure the distance of objects independent of their reflectance (VL53L1X, STMicroelectronics, Geneva, Switzerland).* (Weiner et al., 2019)
        + Capacitance sensors can also be used in this way up to 60mm (Roberts et al., 2021)
      * Temp
        + Used by ARMAR
      * Further research (Roberts et al., 2021)
        + Self healing
        + Triboelectric
        + Damage detection
  + Joint level torque sensors (not for us I dont think)
* Supplementary Electronics (1)
  + Busses and wiring
    - They need to be extremely robust to electromagnetic perturbances. (Stasse & Flayols, 2018)
      * Wire size, types: flexible PCB, ribbon, multistrand, single strands
      * Flat flex cables smallest size is 6 ARMAR-6 (Weiner et al., 2019)
    - Communication lines
      * Servo to servo
      * Ethernet
      * More basic methods
      * Where to put control electronics
      * Protocols (does this go here?)
      * RS422 Kaneko et al. (2019)
      * TS232C Kaneko et al. (2019)
      * EtherCat Kaneko et al. (2019)
      * I2C ARMAR-6 sensors (Weiner et al., 2019)
  + Power requirements 0.25
    - Mostly only important for mobile robots powered by batteries
    - Need to have a constant voltage, complicated by battery charge loss
    - Fuses
  + Temperature sensing
    - Briefly mention cooling systems
* Arm control systems 5
  + State of technology, how good is the control
  + Control electronics

Control of a robotic arm is often split into 2-3 controllers. A faster, higher level controller like a miniature PC deciding on the joint movements, and running more complex tasks like computer vision and machine learning algorithms. This high level controller then sends signals to a lower level controller that controls torque and turns the joints to the right angles. From here either the low level controller signals in turn go to servo motor controllers, or go to motor controllers and position control is done by the low level controller using sensor feedback. There is of course great variance in the options for this setup, and many robots leave out one of these controllers.

For low level controllers one common option is the humble Arduino. The use of this controller appears to be common in arms that use brushed DC servos as actuators such as Prabhu et al (2021), Sakhineti & Jayabalan (2020) and Yang et al. (2019). The vast majority of hobbyist servos are designed to work well with an Arduino so this is a sensible decision for this case. It is very cheap, and the *mega* version has a lot of IO pins, however it is severely limited in terms of its 16MHz clock speed and <16MBytes of memory ref. Provided that the Arduino is only undertaking simple tasks and is backed up by a higher level processor as in Sakhineti & Jayabalan (2020) and Yang et al. (2019) this is of little concern. Though one would need to be confident the Arduino’s limitations would not be a hindrance later in a robotic arm project before committing to using it. Prabhu et al (2021) used an Arduino as the only controller for the entire arm control system, which was controlled via flex sensors in a control glove that fed into the Arduino’s IO pins. This setup was sufficient in this case as the algorithms implemented for control were basic, however it means that more complex controls would be difficult to implement without rewiring in a new controller or rewriting the code to accommodate an additional high level controller. The Arduino also has sufficient processing to implement basic sensor-actuator control for non-servo motors or to control more complex BLDC servo motors, though no literature examples were found and its 5V output signals provide more flexibility than a 3.3V system.

Another sound option for a low level controller is a Raspberry Pi. This operates at a 3.3V standard and all of its IO pins are GPIO pins with PWM output and analog input, as opposed to the Arduino that only has limited PWM pins and analog input pins. The latest version (Pi 5) also operates at 2.4GHz, around 150 times faster than an Arduino, and with 500 times the memory at 8GB. For these reasons the Raspberry Pi is a popular choice and it has enough processing power that it can be used as both a high level and low level controller on one board. The exceedingly capable TOCABI full humanoid robot uses the Raspberry Pi as a low level controller (Schwartz et al., 2022). It is unclear from the paper whether only one is used or multiple are used to control different subsystems in this case, but the higher level control is then done from a embedded PC with an i7-9700 intel CPU. This makes sense in this case as the code for a complete humanoid robot is exceedingly computationally expensive (and high computational power creates opportunities for better robotic capabilities) so a high quality (for the time) PC would be desired. On the other end of the spectrum is a humanoid torso (including arms) designed to be low cost (in contrast to the $100,000+ NZD in parts for TOCABI (Schwartz et al., 2022)). This torso by Hayosh et al (2020) is designed as a “social interaction robot” whose primary purpose is to provide emotional support and social interaction to users. This robot uses a Raspberry pi as the only controller, which interfaces with Dymaxil BLDC servos. Not using an embedded PC in this case greatly reduces costs and simplifies the programing, while the Raspberry Pi still allows for considerable computing power which is enough for most tasks. There is perhaps an argument that an embedded PC would also be beneficial in this case as the social interactions are likely to be autonomous and driven by machine learning (Sakhineti & Jayabalan, 2020), which is computationally heavy, but an embedded PC obviously comes at a significantly increased cost.

Generally low level controllers are either Raspberry Pis or Arduinos, but other microcontrollers are also used such as the the 32 bit Atmel XMega64 microprocessor, 2 of which were used in the sign language robot from Meghdari et al. (2018). Or the custom motor controller board used for the dual arm robot from Shut & Hollis (2019). There are also custom motor-driver-sensor modules such as those used by the 2 armed robot from Asfour et al (2019) and humanoid robot from Kaneko et al. (2019). In these cases some sort of microprocessor will have been used though exactly what is not specified. The advantages of using these microcontrollers would be additional capabilities they may have such as increased current output or true analog output, as well as their decreased cost and size. The disadvantage though is the increased development work required due to a lack of support and libraries, and as the development work is generally the largest cost of a robot (Schwartz et al., 2022), there is a strong argument to avoid these processors in a robotic arm, with the exception being for mass production arms where development costs are less important than unit costs.

In robots that have a higher level controller on top of some amount of lower level controllers, this higher level controller is invariably a miniature PC. This is because if the higher processing power of a PC was not required then the control would simply be handled by a single lower level controller such as in Prabhu et al (2021). One common setup for a miniature PC is to have one or multiple high level controllers (PCs) integrated with many very low level controllers, one for each actuator, that control torque and position for that actuator. The system has the advantage of having very fast processing of data, but it does increase complexity as it means there is a low level controller for every actuator as opposed to one low level controller controlling many actuators. The use of PCs and their increased processing power allows for greater abstraction and autonomy and one robot that embodied this concept is the wheeled, 2 armed robot from Asfour et al (2019). This robot uses four miniature PCs and an additional graphics module for controlling the robot at the 3 levels of abstraction: perception and reasoning, movement/pose execution, and actuator control. This higher processing power allows for the use of a real time OS, for machine vision processing and for complex task planning, however this obviously comes at a significant financial investment. These PCs integrate with the sensors and actuators through EtherCAT (Ethernet for Control Automation Technology) protocol ethernet cabling. To allow the PCs to directly interface with the hardware the motors and sensors have some sort of controller directly attached to them that allows them to use the EtherCat protocol (it is not made clear whether this is custom or off the shelf). This adds complexity to the sensors and actuators, likely increasing costs. Another robot that also uses a similar control system is the humanoid robot from Kaneko et al. (2019). In this design two miniature PCs i7 intel processors (one 5th generation and one 7th generation) communicate via EtherCat to motor drivers, and the motor drivers integrate torque and position feedback from joint sensors. Communication to the cameras, inertial management unit and other sensors is then achieved via USB. In both systems the motors drivers control BLDC motors (presumed on Kaneko et al. (2019)) as opposed to servos. The full humanoid robot from Tatsch et al. (2017) and the testbed robotic arm from Mick et al. (2019) also use a similar system, except that the actuators used are off the shelf Dynamixel BLDC servos that be controlled via a PC over USB (with a protocol converter intermediary).

An common alternative to having a controller for each actuator (whether custom made or embedded into a servo motor) is to have one (or a few) lower level controllers that control multiple motors connected to a higher level PC controller. This is the case in the full humanoid robot from Schwartz et al. (2022) where a miniature PC with an i7 9th generation processor controls 3 lower level controllers: 2 Raspberry Pis and 1 secondary PC. Similarly the robotic arm from Yang et al. (2019) and the dual arm social interaction robot from Sakhineti & Jayabalan (2020) use a PC as a higher level controller and an Arduino low level controller. Other off the shelf microprocessors can also be used as a lower level controller such as in the case of the dual arm robot from Shut & Hollis (2019) which uses one large control board coupled with a Intel Core2 Duo @ 2.4GHz and in the sign language robot from Meghdari et al. (2018) which uses multiple Atmel XMega64 microprocessors. All of these setups offer lower costs than having controllers for every motor if a higher level controller is required for performance reasons, but adds complexity and cost over a single controller such as a PC controlling BLDC servos or standalone raspberry pi.

In most cases the PC is selected to be an embedded part of the robot so a miniature PC is used because of space constraints, as well as because it is an all in one solution. However non-mobile arms that are in the prototyping stage may just use a standard PC or workstation to save on costs, such as may be the case in the arms from Yang et al. (2019) and Mick et al. (2019) (however neither specified what PC they used).

pc->controller

* (Schwartz et al., 2022) - raspberry pi + *embedded PC with an i7-9700 intel CPU running Ubuntu 18.04 and the re al-time operating system Xenomai-3.0.10*
* (Shut & Hollis, 2019) UR-5 style arms on an omni wheel robot - big controller board
* Yang et al. (2019) PC + Arduino
* (Sakhineti & Jayabalan, 2020) SHARLA Torso with wheels and arms for social interactions - unity + arduino

pc->actuators

* (Tatsch et al., 2017) Dimexel full humanoid basic
* (Mick et al., 2019) Pretty arm that is primarily a testbed (servos)

None of previously mentioned boards

* Sign language robot (Meghdari et al., 2018) RASA (Robot Assistant for Social Aims).

## Software environment

The control software used on robotic arm depends on a few factors, namely:

* The limitations of the controller running it. E.g. and Arduino can only run C programs and can’t run an OS.
* The complexity of the tasks to be undertaken.
* The speed at which tasks must be undertaken.
* Software cost.
* How easily the software integrates with the actuators and sensors.
* The desire for standardisation and interoperability.

For the robots that use an Arduino or other microcontroller there will generally be only a single operating environment: some version of embedded C or Object-C, however a few microcontrollers do allow for microPython wrappers or even Javascript, such as the ESP32, Pyboard and Raspberry Pi Pico (Ref). The many of these microcontrollers, the Arduino in particular, also have an extensive range of libraries that could be considered as a sort of middleware.

For embedded PCs like the Raspberry Pi (which is not a microcontroller like the Raspberry Pi Pico) and miniature PC’s there is a much wider range of options for software environments. Firstly is the actual operating system and then there is the middleware.

### Operating System (OS)

The operating system used is generally some version of linux as this allows for stable systems and overarching control by the programmer. For the robotic systems running on a Raspberry Pi these will commonly use the default Raspbian OS, but if running Robotic Operating System middleware (which is not in itself an OS) it is more common to run a version of Ubuntu on the Pi as it has better integration (The Robotics Back-End, 2020). Ubuntu appears to be the most commonly used OS on miniature PCs in robotics. Ubuntu 14.04 Linux is used on the dual arm assistance robot from Shut & Hollis (2019), and Ubuntu 18.04 is used on the full humanoid robot from (Schwartz et al., 2022). However most papers do not specify what operating system is used, as in many cases they are interchangeable and only minor adjustments need to be made to run the application in a different OS (ref?). It is possible that some prototype robotic arms such as the one from Sakhineti & Jayabalan (2020), where the computer only used to communicate small amounts of data to the lower level control board, could run Windows. In this case communication is done via a com port and the computer is simply sending the motions from a digital pre-recording to the Arduino so nothing complex needs to be done on the operating system. It is not specified what OS they use however.

### Middleware

There are extensive options for middleware specifically designed for robotics. Middleware is an extension of the operating system’s functionality. In robotics this extension commonly includes such features as support for real-time systems, physics simulation libraries, communication protocol support and integration with machine learning libraries. Generally most complex robots will use some version of middleware

The most common of these middlewares is ROS (Robotic Operating System), despite the name it is not a standalone operating system but rather a middleware primarily used on Ubuntu, although according to the distributors of the software (Open Robotics, 2020) support for other linux systems and Mac OS X also exist. Both ROS 1 and ROS 2 exist and they are open source projects supported by the commercial company Open Robotics (Open Robotics, 2020). ROS 1 has been criticised as lacking support for real time operations (Stasse & Flayols, 2018) but is commonly used across the board in robotics applications (Macenski et al., 2022). ROS 1 is also lacking in terms of security and reliability (Macenski et al., 2022). ROS 2 is an alternative that was first developed in 2017 to combat these challenges. ROS 2 is fully supported by Ubuntu, Mac OS and Windows 10, and has improved security using the Data Distribution Service (DDS) standard, which is used in such high stakes applications as spacecraft. Importantly it also provides real-time support, which is critical in fast moving, accurate robots. There are few examples in the literature of ROS 2 being used but this is likely as it is such a new software. In contrast, examples of ROS 1 being used include in the medical teleoperation arm from Procter and Secco (2021), and the full humanoid robot from Pal Robotics (Aller et al., 2022). There is a strong argument for the use of ROS in robotics as it allows for programs to be easily built on previous work due to it being open source and the most popular middleware (Saeedvand et al., 2019).

An alternative to ROS is OpenRTM (Open source Robotic Technology), which is heavily oriented towards real time operations (Stasse & Flayols, 2018). This is commonly used in Japan, and is used in the Humanoid Robotics Project competition there (Stasse & Flayols, 2018). It comes with realistic simulators and good support for balancing and walking.

YARP (Yet Another Robotics Platform) is, yet another, middleware that can be used for robotics (Stasse & Flayols, 2018). The main use of YARP is to allow for straightforward communication across multiple pieces of hardware and operating systems (Stasse & Flayols, 2018) and for this reason it allows for the use of many different communication protocols (Macenski et al., 2022) and runs on Linux, Mac OS and Windows (Stasse & Flayols, 2018). It is most commonly used on legged and humanoid robots (Macenski et al., 2022).

These are of course not the only options for robotic middleware, there are also frameworks in Matlab/Simulink and an MCA (Modular Controller Architecture) middleware that are useful for the control aspects, and middlewares such as Orocos and which are similar to YARP and ROS in that the provide many low level communication features (Vahrenkamp et al., 2015). There are also middlewares designed for specific actuator-sensor systems such as Pypot, a python library for controlling Dynamixel motors and sensors. Such a system would normally have little relevance except for the fact that Dynamixel motors are used in so many robots. Not all of these robots use Pypot but some do such as the testbed robotic arm from Mick et al. (2019). The use of python here on a test arm is not an issue, but it should be noted that running a production level or complex robot primarily using the python language and a library such as Pypot would be inadvisable due to python's very slow execution speed. Another option is to develop a custom middleware such as the ArmarX middleware developed and used for the ARMAR series of full humanoid robots (Asfour et al., 2019). However this option comes at considerable expense/resource investment, and comes with the downside of limiting software interoperability (though this could be an advantage for commercial robots). Other robots may use a real-time system (but not a robot specific one) such as Xenomai-3.0.10 which is run on Ubuntu 18.04 in the full humanoid robot TOCABI (Schwartz et al., 2022). Finally, the other option is to use no specific middleware, which may be the simplest option when the robot is not complex and there are a limited number of sensors and controllers. This is likely the case in the social interaction torso robot from Sakhineti and Jayabalan (2020) which only uses an Arduino that is communicated with via a com port, and is of course the case when the robot is controlled entirely by microcontrollers such as in the control glove controlled arm from Prabhu et al (2021).

## Software control methods

### Control Input

The software for controlling a robotic arm can be broken down into hierarchical components, depending on what the control input is to the arm. There are many input methods that are minimal and the arm has a high degree of autonomy. For example where the robot is making decisions about how to best pick up an object, or how to climb a ladder etc. This is an important area of research but the focus of this project is on the hardware of the arm and the development of a flexible software/firmware base, so that further work can develop these autonomous methods. Other input methods do not need the robot to make many decisions about joint angles as the input controls have an almost 1 to 1 correspondence with the robot DOF. Examples of these methods include control gloves (for example in the robotic arm from (Prabhu et al., 2021)) and machine vision systems for teleoperation. If well designed for interoperability then all of these input methods can be used to control a general arm model that can change based on the hardware, and that is the aim of this project: to develop a robotic arm and the base software for controlling said arm such that it can be easily implemented with a variety of control methods at a later stage.

Doing this involves implementing a model of the arm in an easy to use software that can control the real world arm. An example of this is the dual arm robot from Sakhineti & Jayabalan, (2020) that is controlled by a Unity 3D GUI with a joystick controller. This robot allows for play and record functionality, but could easily be extended to do more complex functions. A similar digital model was also used for the arm from Yang et al. (2019), where a Xbox Kinect skeleton framework coupled with the Open Dynamics Engine physics simulator are used as a digital twin of the arm.

### Low level control methods

Any robot needs some level of low level control over the joints in order to get them to the desired position at the desired speed, without exceeding safe torques and potentially while interacting with objects. Arguably the simplest and most well established method for low level optimization control are PID controllers. Within this category there are also many variations such as PD controllers, PI controllers and fractional order PID controllers (Muñoz et al., 2019). If the robot is using servo motors then position control is already handled by PID controllers in the servos (ref), and more expensive BLDC servos may also have speed or torque control. Otherwise a feedback loop is implemented between the angle sensors installed, any torque sensors and the inputs to the motor. Motor inputs could include PWM duty cycle, current, or operating frequency depending on the type of motor and driver. Another similar method is a fuzzy-tuned PI control method which was found by Yin et al. (2022) to considerably decrease the speed fluctuations and increase the control performance vs a PI controller on a humanoid driven by tendons. In addition to the array of different methods mentioned there are also different methods for tuning the parameters of the controller. For a simple PID controller there are well established methods such as Trial and Error, Zieglar-Nichols Step response, Relay Tuning, Cohen-Coon (Borase et al., 2020). There are also less common methods such as neural network tuned PID controllers which were shown by Ali Talib Jawad et al. (2021) to improve control. Some common approaches to tuning fractional order PID controllers are Monge's Method, the Artificial Bee Colony algorithm or the Counter-slope method (Muñoz et al., 2019), though a novel graphical method that shows robust (and comparable) performance for a robotic elbow joint is also presented by Muñoz et al. (2019). Other low level control methods also exist for robotic arms but there is little literature on the use of low level control in humanoid arms to base this on.

### Higher level control

A major aspect of controlling a robotic arm when the control input is not exact in its specifications for the position of every joint (e.g for teleoperations control) is in determining how best to angle the joints of the arm to place the end effector in the desired position (Mick et al., 2019). An example of this is when a robot is programmed to pick up an object and must place the end effector around the object. This is commonly referred to as the inverse kinematics problem (Mick et al., 2019). Typically there are libraries that come with robot middleware for solving these inverse kinematics problems both numerically and analytically (Mick et al., 2019) however robots with 7 degrees of freedom or higher will typically need their inverse kinematics to be solved numerically due to the complexity of the equations and processing involved (Kamil Khusnutdinov et al., 2019). One common such numerical method is local optimization (Mick et al., 2019) and varies libraries exist for this such as the IKPy (Python) library used for controlling the testbed robotic arm from Mick et al. (2019). Neural networks are another approach to the inverse kinematics problem and were to be slightly superior over local optimization when tested on the same robotic arm (Mick et al., 2019). Neural networks also offer the opportunity to add in bais to the training data to make the robot select kinematic solutions that are more useful in the real world (Mick et al., 2019), for example having the robot face the palm down when manipulating objects up high.

PID tuning methods…

Look at grasping methods and control, applications of PID

Can add a bit on kinematics/control but remove its removal from scope

* + - Programmable controller and interfacing
      * Add below in for more references
      * Often require top of the line computing for a robot with legs (Stasse & Flayols, 2018), for instance *the HRP-4 robot from Kawada Robotics is using a Pentium M at 1.6 Ghz which is not equipped with the Intel Turbo Boost technology in contrast with processor which can be found in the REEM-C humanoid robot.*
      * *most of the humanoid robot developers tend to equip their humanoid robots with different kinds of small processing units such as mini PCs (especially the robots meant for competition purpose) (Allali et al., 2016; Huan et al., 2016; Saeedvand et al., 2017; Ficht et al., 2018). On the other hand, in addition to the hand-made controlling boards, commercial controlling boards are used extensively. In this way, Raspberry Pi boards (Mejías et al., 2017), ARMbased boards (Almubarak & Tadesse, 2017), compatible Arduino boards (Al-Busaidi, 2012), or off-board computing controls (Khokar et al., 2015) are commonly used on humanoid robots. (Saeedvand et al., 2019)*
    - Arduino
    - Rasberry Pi (Ghael et al., 2020)
  + Operating system
  + Middleware
    - ROS most common but also YARP and OpenRTM (Stasse & Flayols, 2018)
    - Same ref, runs on ubuntu well, but can run on other systems
    - ROS
      * Lacks some real time capabilities, most common is OROCOS with Xenomai, BSD
      * More at (Saeedvand et al., 2019)
      * ROS 2
    - OpenRTM
      * More oriented towards real time applications
      * Mostly Japan
    - YARP (yet another robot platform
  + High level/input
    - Cooperative
      * Is this much different?
    - Digital twin
  + Low level positional and torque feedback
    - PID
      * Tuning
        + Neural network
        + IWO - invasive weed optimization
    - Fractional order PID
      * Tuning methods
        + *(Muñoz et al., 2019)* (Presents a new method, but also compares to classical methods) [*https://www.worldscientific.com/doi/pdf/10.1142/S0219843619500427*](https://www.worldscientific.com/doi/pdf/10.1142/S0219843619500427) *Many of them are based on the numerical solution of nonlinear equation systems.6,8,10,11 Other approaches based on optimization methods can be found in the literature such as Particle Swarm Optimization algorithm (PSO),7,13,15 Arti¯cial Bee Colony algorithm (ABC),16,17 Fire°y Algorithm (FA),18 or Di®erential Evolution method (DE).19*
    - Fuzziness
    - Feed forward control
  + Inverse Kinematics
    - * Complicated as shit?
      * <https://www.sciencedirect.com/science/article/pii/S0094114X21003700?casa_token=aS3xMY6lO6sAAAAA:Tfk7IrUdQrDS-70s9YgrBf8Lo9h8t2K1-7YlDSwK2AUWlo5jrDavDuGN2yCEEwL0oqk_54CoJ9IQ>
      * *(Mick et al., 2019) Determining a set of motor angles that put a robot's endpoint at a target position in its operational space is a common problem in the field of robotic arms, and is usually referred to as the Inverse Kinematics (IK) problem. As it comprises seven independent DoF, Reachy typically displays kinematic redundancy, implying that there is an infinite number of distinct solutions to this problem for each reachable target position. Thus, in order to drive the robot's endpoint position to a given target, one needs to determine which set of angles to apply, among the infinity of possible sets. However, the numerical expression of this under-constrained geometrical problem is non-linear, which makes analytic solving impractical and costly in terms of computation.*
      * *Currently, there are libraries that allow programmatically solving the problem of inverse kinematics. They provide both an analytical solution [9] and a numerical one [2]. B*
      * 7 or higher DOF then need numerical solution (Kamil Khusnutdinov et al., 2019)
      * Lots of plugins exist (Kamil Khusnutdinov et al., 2019)
    - Local Optimization (Mick et al., 2019)
      * *Instead, a widespread method used by roboticists to solve IK problems is to employ local optimization. This method relies on a cost function, attributing a scalar value to any set of angles to quantify to what extent it is a good solution with respect to the IK problem: a lower cost means a better solution. Usually, this cost function is based on the distance between the target and the endpoint position, which can be analytically determined with the geometrical model of the robot. Then, through a step-by-step process, the optimization finds and returns a local minimum of this cost function, that should correspond to one of the sets of angles putting the endpoint at the required position.*
      * *We used the Python library IKPy (Manceron, 2015), a generic IK solver*
      * *A code sample showing how to use IKPy with Reachy is available online*
    - Supervised learning NN (Mick et al., 2019)
      * Was a bit better than local optimization
      * More computation
      * Small python network
        + *feed-forward multi-layer perceptron including two fully connected hidden layers of, respectively 64 and 128 neurons*
      * Need to record angles on all joints using embedded sensors and get a dataset
      * Can introduce position bias based on training data
      * *Regarding network structure, we noticed that adding more hidden layers or increasing their size does not draw significant benefits and can even result in the network overfitting the examples*
  + Conclusions - 0.5
  + Calibration
    - Errors could be accounted for by running calibration programs and adding offsets (Mick et al., 2019)
    - <https://ieeexplore-ieee-org.ezproxy.waikato.ac.nz/document/8637801>
  + Motor drivers
  + Safety
    - *all robot actuators go immediately to an e-stop, where they hold their current positions with reduced torque.* (Asfour et al., 2019)
* Overview of control input methods 1.5

The focus of the envisioned robotic arm project is not on the input method for controlling the arm, but whether for future work on this arm, or for the project’s real world relevance, the arm should be designed with an awareness of popular uses and control methodologies. In this way real world relevance can be established and design choices will not be made that would compromise the arms possible use cases.

* + Why?
    - Because being aware of the main control methods allows for the development of a hand that can be controlled more easily should a control system be implemented in the future and makes the research more relevant to a wider range of disciplines
  + Brain controlled
  + forearm/muscle controlled
  + Control gloves
  + Vision systems detecting human hand movement
  + Autonomous
  + Digital twin
  + Dials etc

Hardware

*Moreover, the robot's arms provide standardized interfaces, which comply with the International Standards Organization's 9409-1-50-7-M6 to quickly exchange end effectors (e.g., different hands or grippers), assuming they are 48-V compliant and have an EtherCAT interface. (Asfour et al., 2019)*

*Tocabi has an ISO 9409-1-50-4-M6 end-effector (Schwartz et al., 2022)*

(Saeedvand et al., 2019)

