Timothy Blake: Assignment 2

October 21, 2024 9:26 PM

Timothy Blake [2442745]

This is my COMP 444 portfolio, Assignment #2. It is an extract of unit 3 and 4 readings, robotic arm design, food for thought exercises, instructor exercise and a forum posting I made on Arduino IDEs.

I strongly recommend reviewing my GitHub as all code, and supplement example work (designs, videos, etc) will be hosted there. https://github.com/malcolmsdad/COMP444/

Thanks!
Tim

Unit 3 - Introduction - Robotic Movement 2 – Affectors

October 17, 2024 8:07 PM

Affectors are the key to allowing a robot to interact with its environment. Affectors affect the environment by touching, grabbing, and other motions. In this unit you will examine different affectors in robotics.

Learning Objectives

After completing this unit, you should be able to

- describe various types of affectors, both how they are created and how they work.
- discuss the creation of robots that employ affectors to interact with their environment.

Readings

Please read the following chapters in the textbook:

- Grasping at Straws (Chapter 6)

Questions to Ponder

At the end of each chapter in the assigned readings there are questions labelled “Food for Thought”. Please answer these questions as best you can in your weblog, which will become part of your portfolio of competence submitted for marking during this course.

Exercises

Exercises for this unit can be found in the *Instructor’s Weblog* on the Landing. Please follow along with the exercises and programs using your own Arduino kit, and keep a record of your explorations in your own weblog.

Further Readings

At the end of each chapter in the assigned readings you will find a section titled “Looking for More.” While the links and readings mentioned in this section are not assigned, please feel free to examine them if you are interested, or ask questions on the Landing.

From <https://comp.athabascau.ca/444/r1/unit03.html>

Unit 3 - Instructor Blog Entry

October 17, 2024 8:27 PM

This unit completes the section on robotic movement. By now you should have completed all SIK Guide tutorials and circuits from Circuit #1 through Circuit #12, inclusive.

Design Question: How many motors, and of what type, would you require to make a fully functional robotic arm that had a working elbow, wrist, and end affector (i.e. a simple clamp)? What components would you add if you wanted the clamp to be able to tell how hard it was grabbing an object such as an egg (i.e. to avoid crushing it)? Discuss your design in your weblog in detail, especially describing the choice of motor for each joint, the degrees of freedom and the range of motion.
(See: page 63 of textbook)

Programming/Circuit Task: Since we don't have all the hardware to build a robotic arm, imagine you have been given the task of creating the elbow joint. Select the appropriate motor for this task, and then create a program and circuit using your Arduino which can demonstrate your motor performing the correct elbow movement. It may help if you tape an object such as a popsicle stick, drinking straw, or long skinny piece of paper to your motor to demonstrate the movement of the lower portion of the arm under control of your program. Your program should take as input a number of degrees to move the elbow from an arbitrary starting position. For example, if you choose 'fully straight' as the starting position, this will be designated 0 degrees (start). The arm could then bend about 170 degrees, indicating 'fully bent' (check the amount of bend on your own elbow from hand straight out to hand near your shoulder for reference).

As always, keep detailed notes in your weblog of the entire development, testing and debugging process. If you have video capabilities, upload a video of your completed project. You should also include a listing of your code in your weblog, as well as a description of the final wiring of your project.

Design: Robot Arm

October 17, 2024 11:32 PM

How many motors, and of what type, would you require to make a fully functional robotic arm that had a working elbow, wrist, and end affecter (i.e. a simple clamp)? What components would you add if you wanted the clamp to be able to tell how hard it was grabbing an object such as an egg (i.e. to avoid crushing it)? Discuss your design in your weblog in detail, especially describing the choice of motor for each joint, the degrees of freedom and the range of motion.
(See: page 63 of textbook)

I can think of three possible ways for a robot to be able to pick up an object without crushing it, with a minor variance only at the end. The primary issues to over come is a term we learned in this chapter called manipulation problems and how to identify grasp points.

If a robot has an immersive AI, then a simple image recognition of the object, could allow the robot to search up typical physical attributes of an object (like an egg for example) including it's weight, ability to estimate center of gravity, the shells tensile strength etc. A properly programmed robots internal state should allow the robot to know how strong it is and how to a range of force. Ie: If the robot applies 12 watts to an actuator, it impose 12 ft/lb if the robot applies 5 watts, its only 5 ft/lb. So please be gentle!

Another option would be to rely on trial and error. Have the robot attempt to pickup the object to only a very short height, with increasingly greater grip strength, or lower grasp point, until the robot can successfully pick it up without dropping. This is a dumb robot, so it never remembers that, just keeps trying.

Finally, I suggest a specifically designed endeffector for the task. If we know that the task of the robot is to pickup eggs, lets design it that way. Specifically, an endeffector that matches the shape of an egg, is designed with firm but gentle method of grasping it - perhaps some malleable or elastic medium is what's in contact with the egg, but that is mounted on a more firm portion which is moved by the actuator.

Goal:

- To design a fully functional robotic arm with elbow, wrist and clamp
- Able to pickup an egg without breaking it
- Simple design minimal components, maximum reliability

Design:

3D Model:

My drawing and artistic skills are dismal at best, so I'll use my previous Sketchup experience to draft a design of the robot arm. I'll do my best, but my experience is drafting decks and building plans with dimensional lumber, not robots with servos and links!

- STL Format
- SKP Format

Overview:

- Complete Diagram
- Complete Diagram (alt. view)
- Claw Details

Components, Joints and Motors:

Labelled Diagram

Use this diagram to identify the letter below:

- A, G: Shoulder and Wrist Joint
  - 180 degrees rotational
  - Servo
- B, D, F: Shoulder, Elbow and Writs Joints
  - 90 degrees prismatic
  - Hydraulic pump
- C, E: Arm Link
- H: Slider prismatic joint
  - Slider causes clamps to open or close grip
- I: Cylinder link
  - A central cylinder a for claw prismatic joint, and a ring that slides along the cylinder (prismatic) moved by hydraulic pump
- J: Claw connecting link
  - Static link transfers prismatic to rotational
- K: Claw mount
  - 45 degree rotational
  - Moved by sliding action, hydraulic pump
- L: Claw link
- M: Elastic Band link
  - The bands are the only part that comes in contact with the egg, and provide extra grip through friction.

Joints:

Claw - Rotational
Elbow - Prismatic
Ring - Prismatic
Ring - Rotational
Shoulder - Prismatic and Rotational
Wrist - Prismatic
Wrist - Rotational

Degrees of Freedom:

Calculation Worksheet
Total DOF: 8

Circuit: Elbow Circuit

October 17, 2024 11:34 PM

For my robots elbow, I decided to forgo the rotational joint, as the two other rotational motors (wrist and shoulder) should allow for enough range of motion and freedom of movement.

Reviewing the elbow again:

Elbow - Prismatic

I've selected a hydraulic motor (piston) to affect upon the arms in relation to the shoulder. The primary action of upper arm and shoulder is to lift, meaning the motor selection must be very strong.
Hydraulic motors are very powerful and very reliable. They have limited range of motion, as the pivot points and piston impost limits. The elbow will allow rotation of 0 to 170 degrees.

Not displayed is the hydraulic pump, or hydraulic lines.

Goal: Write a program that inputs a number of degrees and moves bends the elbow that much.

Problem: Converting degrees (rotational) to distance.
https://github.com/malcolmsdad/COMP444/blob/main/Excercises/U3%20-%20Elbow%20-%20Trigonometry.png?raw=true

We want to be able to move the arm by specifying degrees "D", but since we're using a piston along "C" then we must calculate the length of C to move. Unfortunately, that also means that we need to know the values of A and B. I'll assign const values for those in the code.

A quick google, and I was able to find this calculation " Law of Cosines " to calculate the length:
$c^2 = b^2 + a^2 – 2ab \cos \gamma$

I used an example circuit where a servo was controlled by a potentiometer. I created a new class to represent a piston controller, it allows for reading current length of the piston, and a function to move the piston. The code reads the value of the potentiometer, converts to degrees, then using the trig function above, to calculate the desired length of the piston. The code then reads the current length of the piston, and move it by the difference of desired and current.

Here is the code here:
https://github.com/malcolmsdad/COMP444/blob/fde853753d40ff637935de238e4d858a6c029d60/Unit%203/Instructor_Excercise____Elbow_Circuit/Instructor_Excercise____Elbow_Circuit.ino

Unit 3 - Chapter 6

October 17, 2024 8:12 PM

Chapter 6 is entitled Gasping at Straws / Manipulation. I suspect this will be a complicated one. Immediately we're introduced to the term manipulator, and it being a effector. It can be any type of gripper, hand, arm or body part that is used to move things. Manipulators typically consisted of 1+ links connected by joints, and the endeffector. Endeffector is like the fingers on a hand - does the actual touching of objects in world.
Endeffectors relies on manipulator to move itself into position for the endeffector. In this case, manipulation involves entire body. Arm must move to make up for its own joint limit. We then learn that the endeffector is limited by the arm, that is limited by the joint limit and its free space.

Then we move to Teleoperation (introduced in Chapter 1) and the difference between teleoperation and remote control. We also learned about the challenge puppetering (a form of teleoperation), and its inherent difficulties: complexity, interface constraints and limitation of sensing.
Teleoperation has been used in surgery with great success. Finer, smaller controls, smaller incisions, reduced heal time and risk to patient. One suggestion to overcome some of these difficulties are exoskeletons, which are more of shells on top of a body, however they are not considered robots.

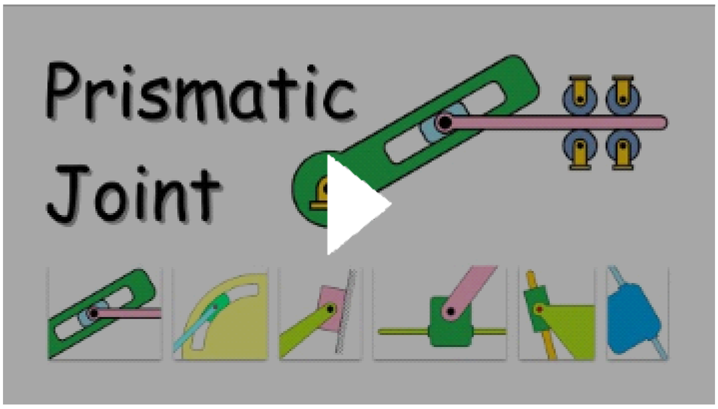
And now on to why manipulation…
First we learn about kinematics (correspondence between actuator motion and resulting effector motion), then joints (connections between links), as well as types of joints (rotary or prismatic).

Robotic manipulators have one or more type of joint, and each join typically provides on controllable DOF, requiring multiple actuators. Materic contrasts the many required components for a robotic arm vs a human arm with seven DOF. Additionally, ball and socket joints are very difficult to recreate in artificial systems.
Mimicking the appearance of a humanlike shapes is called anthropomorphic and is extremely complicated and associated manipulation problems.

From there we moved to defining what is required to control a robot manipulator is called kinematics, including describing manipulation problems (what is attached, how many joints are there, how many DOF) and how to accomplish to move the manipulator to accomplish the goal (inverse kinematics). Further to that are dynamics which describe of an object that has motion or energy.
Managing kinematics, dynamics, compliance within an environment and grasping an object are the reason that manipulation is so hard.

Note:
While doing the textbook exercises, I found that my understanding of Degrees of Freedom was lacking, especially while trying to calculate DOF. I re-read the textbook but unhappy with my comprehension so I watched this video:
[Understand prismatic joint and how it affects the degrees of freedom in a system]

I found it very helpful, and I sure hope it's accurate, because I intend on using the algorithm:
DOF = 3xB - 2xJ
(B = # of Bodies, J = # of Joints)


Prismatic
Joint

Definition

October 17, 2024 8:13 PM

Manipulator: An affector. Any appendage that is used to affect and move objects in the environment.

Manipulation: A goal driven movement of any type of manipulator.

Manipulator Links: individual components of a manipulator that are controlled independently. Ie: upper arm and lower arm.

Endeffector: The endeffector is the part of the manipulator that affects the world. Fingers on a hand, foot of the leg etc.

Joint Limit: The extreme limit for how far a joint can move.

Free Space: The space in which a movement is possible

Teleoperation: Controlling a machine at a distance

Remote Control: Controller simple mechanisms

Robot Assisted Surgery: A surgeon remotely controls a robot that moves inside the body of the patient to cute and suture.

Exoskeleton: A structure that a human wears and controls. It provides additional strength, or mimics controls of a robot for teleoperation.

Kinematics: Correspondence between actuator motion and resulting effector motion

Joints: The various links of a manipulator are connected by joints.

Rotary Joint: Ball and socker join, providing rotational movement around a fixed axis.

Prismatic Joint: A piston: providing linear movement

Anthropomorphic: Human like shape

Manipulation Problems: Problems about where the endpoint is relative to the rest of the arm, and how to generate paths for the manipulator to follow in order to get the job done.

Inverse Kinematics: The conversion from Cartesian position to an endpoint and the angles of the whole manipulator

Dynamics: Properties of motions and energy of a moving object

Grasp Points: The point on an object where fingers need to grasp relative to the center of gravity, friction, obstacles etc.

Compliance: Yielding to the environment required for tasks with close contacts such as sliding along a surface.

Food for Thought

October 17, 2024 8:12 PM

How many DOF are there in the human hand? Can you control each of them independently?
I will attempt to identify each joint within the hand, describe and count their degrees of freedom.

Each finger and thumb has three joints:
1. Where digit is attached to hand has 2 DOF: up, down, left, right
2. Two knuckles on each digit with 1 DOF: up and down. All the joints within a digit allow for the entire digit to translate forward and back, adding another DOF

The total DOF of a human hand is 25 - although.
Not all DOF are independent however, simply due to the anatomy of a hand, some joints or ligaments can be tight or tied closely to another. For example, its extremely difficult if not impossible for me to make a fist, while leaving the ring finger extended.

Which of the two joint types we discussed, rotational and prismatic, is more commonly found in biological bodies? Can you think of examples of specific animals?

Rotational joint type is most commonly found in biological bodies, in the form of a ball and socket. Typically found in complex joints where large links are attached to bodies, such as shoulders, hips, knees. They're extremely difficult to reproduce in robots.

Are astronaut suits exoskeletons? What about a lever-controlled backhoe?
Yes, astronaut suits are exoskeletons. An exoskeleton, in bioloigcal systems is the form of a hard outer structure, such as s shell, that provides protection or support. This is exactly what an astronaut suit does. It provides heat, or cooling, oxygen to breath, communications with other astronauts and support team. Some space suits, when equipped, provide locomotion in the form of a thruster pack.

A back hoe is an interesting question. It does provide some protection, and ofcourse increases strength. It provides locomotion as well as comfort (such as AC or heat). Many backhoes have a feedback option, where you can feel the substrate that the bucket is cutting into, and they definitely have sensors, including modern GPS, temperature, etc. The only question that I have issue with is the part of the definition where the backhoe provides a hard outer surface. It definitely does that, but in robots the definition states they must 'wear'. I don't consider the operator of a backhoe wearing the machine, so no. I don't believe a backhoe is an exoskeleton. The machine Ripley used in Aliens is definitely an exoskeleton though!

Imagine an exoskeleton which has its own sensors, makes its own decisions, and acts on them. This is definitely a robot, yet it is also controlled by a human and attached to the human's body. Can you imagine where this could be useful?

Absolutely, and I can provide an example of where that could possible come in handy. During early days of exploration, there was an intense race to achieving the hundreds of 'firsts' in space. The USSR beat the US to almost all of these achievements, however they did with much greater rush and risk. One such event was the very first space walk. Performed by Alexei Leonov, he successfully left the Voskhod 3KD space craft and entered empty space for approximately 12 minutes. The whole event was filmed and touted as a propaghanda win for the USSR. However, it was nearly a complete disaster, Leonov found that as soon as he left the capsule, his suit ballooned up much larger than the crafts door. He had to vent his suit to open space to be able to fit in, resulting in suffering the bends. Now imagine if something went wrong with Bruce McCandless during the first untethered space walk? He achieved untethered EVA by relying on the space suits backpack. Now imagine if a fault occurred that left him unconscious or disabled in another way. If that space suit had the ability to recognize that fact, it could return him safely to the shuttle. Additionally, I could see drone

exoskeletons used to extract personnel from dangerous or war situations.
How about disabled individuals, exoskeletons could allow someone to walk, while the system manages all balancing efforts with internal sensors, decisions and actions.

Robotic dogs have been used to play soccer. Because they are not designed for that task, they move slowly and have trouble aiming and kicking the ball (not to mention finding it, but that's not part of manipulation; it belongs to sensing, discussed in the next few chapters). Some of the best robot soccer teams have come up with a clever idea: instead of having the dogs walk as they normally do, they lower them so that the two rear legs are driving and steering the robot, while the two front legs are on the ground, used to grasp the ball and kick it, which is much easier. This turns out to work very well, even if it looks rather silly. What are the endeffectors of the dogs in that case? Are the dogs now purely mobile robots?

Endeffectors are the part of the manipulator that affects the world. In this case the front legs are endeffectors. Since the wheels do not interact with the ball, or anything else, and are more related to movement than manipulation, they are not endeffectors.
Yes, these are robots. They have a goal (to find the ball and kick the ball), they are aware of their environment, they exist, and can do so by itself. Bend it like the little bot!

Unit 4 - Introduction

October 17, 2024 8:10 PM

Unit 4: Robotic Sensing

Sensors allow a robot to sense what is happening in the world around it. Sensors are critical to creating robots that can interact with the environment by providing a way for the robot to know what its actuators and affectors are doing. In this unit you will examine various sensors and how they can be used to create robots that can perform useful functions.

Learning Objectives

After completing this unit, you should be able to

- describe the various kinds of sensors available to robots, from simple to complex.
- describe how sensors can be used to enable robots to perform their functions effectively.
- describe switches and light sensors and position sensors in terms of applicability to robot sensing.
- describe some of the complexities in creating useful vision sensors.
- discuss the creation of robots that employ sensors to perform useful functions.
- employ the Arduino and the SparkFun Inventor's Kit to create circuits which use various sensors to interact with the environment.

Readings

Please read the following chapters in the textbook:

- What's Going On? (Chapter 7)
- Switch on the Light (Chapter 8)
- Sonars, Lasers, and Cameras (Chapter 9)

Questions to Ponder

At the end of each chapter in the assigned readings there are questions labelled "Food for Thought." Please answer these questions as best you can in your weblog, which will become part of your portfolio of competence submitted for marking during this course.

Exercises

Exercises for this unit can be found in the *Instructor's Weblog* on the Landing. Please follow along with the exercises and programs using your own Arduino kit, and keep a record of your explorations in your own weblog.

Further Readings

At the end of each chapter in the assigned readings you will find a section titled "Looking for More." While the links and readings mentioned in this section are not assigned, please feel free to examine them if you are interested, or ask questions on the Landing.

Assignment 2

It is time to submit your weblog for Assignment 2.

From <https://comp.athabascau.ca/444/r1/unit04.html>

Unit 4 - Instructor Blog Entry

October 17, 2024 8:31 PM

Activities using the text Companion Workbook: Read through the text companion workbook section titled Sensors. Answer the questions posed in this section of the workbook in your own weblog. You will not be able to do the exercises in the workbook, but compare what they are doing with the iRoomba to robot kits like Lego Mindstorms. How do they compare? How do the sensors used in the iRoomba platform compare to the sensors in your Inventor's Kit?

I am unable to, I do not have the Companion Workbook.

I do however, own a iRoomba and can try to estimate the sensors contained within.
My iRoomba is pretty basic vacuum with WiFi control. Mine is not capable of mapping or planning its route, it simply drives until it detects an obstacle or fall risk, or areas with increased dirt. The robot is also aware of its own battery level and status, dust bin volume and air filter status. This indicates a combination of internal (proprioception) and external state sensors (exteroception).

External sensors:
Collision:
- Distance: The primary sensor is collision detection, most likely a distance sensor. The robot runs at an increased speed over unobstructed ground, however as it approaches an obstacle, like a wall or furniture, the robot slows before making contact.
- Contact: The large plastic half circle encompasses the front half of the robot, this gently detects objects in the robots path, so it may adjust its trajectory.

Fall Risk:
- Distance: The robot must be equipped with a downward facing distance sensor, as the robot is capable of detecting fall risks, such as stairs going down.

Dirt:
- Reflective Optosensor: The robot is able to detect areas of extra dirt of dust, this can be achieved using infrared reflectance. An infrared diode and photodiode detect dust by detecting light reflecting off dust and dirt particles.

Dock Location
- Photodiodes: The robot is capable of locating the base dock and returning to it on its own. I believe that the robot is able to do this using infrared triangulation. The dock emits and an IR blast, which the robot is attempting to locate and drive towards. There is a small protrusion on the top of the roomba containing transparent plastic. That plastic captures the IR blast and directs it onto photodiodes. The robot uses the current passing through the diodes to determine the dock locations and navigates appropriately

Internal Sensors:
Battery Status:
- Voltage sensor: can be used to determine what the battery is currently doing (charging, discharging, idle) as well as its level. When the battery is low enough, the robot returns to charge

Filter Status:
- Tachometer/rotational speed sensor: A small fan inside the iRoomba creates the negative vacuum and pushes the air through a filter. The voltage to that fan remains constant, however as the filter fills, resistance to the fan slows the speed of the fan. When the rotational speed hits a limit, the filter is considered full.

Total Sensors:
- Distance sensors: 2
- Contact: 1

- IR Photodiode: 4
- Voltage sensor: 1
- Rotational speed sensor (ie: shaft encoder): 1

The SparkFun Inventors Kit contains an example of almost all of these!

The SIK includes a photoresistor and a distance sensor that could serve the purpose of the two distances sensors found in the robot. SIK includes 4 buttons, and when covered with a large half circle of plastic, could serve as a great collision sensor.
The SIK includes a number of LEDS, however it does not have a photodiode. From my research, I determined that photodiodes and photoresistors are very similar in that they react to light conditions, however photodiodes are much more reliable and react faster than photoresistors. (Quarktwin, 2023) Voltage resistors are built-in to the RedBoard, so no problem there. Finally a rotational speed sensor is included in the robot, which on the surface doesn't appear to exist within the SIK, we could attempt to make on however I wouldn't suspect it'd be very reliable. If we mount a stationary photoresistor next to the fan blades of the iRoomba, the photoresistor may pick up the light blocked by the passing fan blade. However photoresistors are known for their slow performance, leading to an inaccurate operation.

At any rate, for a lower cost and very user friendly kit, the SIK surely has enough for many beginner robotics enthusiasts.

Unit 4 - Chapter 7

October 17, 2024 8:17 PM

What's Going ?
Sensors

In order for a robot to properly operate it must be able to detect its internal and external states. There are two types of sensor proprioception and exteroception, combined they form the perceptual system. Sensors in systems aren't easily read, and uncertainty can be caused by issues like:

- Sensor noise
- Limitations
- Effector/actuator noise and errors
- Hidden and partially observable
- Lack of prior knowledge / context

The chapter then dives in to the types of data that a sensor retrieves, including data types (bits, pixels), and states that simple switches only provide information, further processing is required to determine what to do with that information. Simple sensors cannot provide enough information to reconstruct the world.
Complex sensor can provide more of that information, enough to answer both questions: Given that sensory reading what should I do? And Given that sensory reading, what as the world like when the reading was taken?"
One of the more fundamental challenges is signal-to-symbol problem, meaning ability to render an intelligent response from a sensors output.

Processing sensor information is the next section the chapter focuses. Electronics are used to measure voltage through circuits which is required to read sensors. Sometimes sensory data must be processed to get intended data, this is called signal processing. Computation is the process of extracting meaningful information from an image. Aspects of computation are then described including the required components (wires, electronics, processing, etc) as well as mentioning some specialized hardware like graphics chips.
Perception requires: sensors, computation and connectors.

Types of perceptions are then described:
Action-oriented perception: use the knowledge about a task to look for stimuli and respond.
Expectation-based perception: use knowledge of environment to guide and constrain sensor data
Task-driven attention: Direct perception is where the robot moves to the direction of information
Perceptual classes: World partitioned into categories, meaning information can be more manageable

Some different techniques for sensing the surrounding world, such as using human body temperature ranges to identify humans, or if everything is static, the human is what's moving.
Examples of how different sensors can accomplish the same task.
Another method of using sensors is combing multiple sensors together which is called sensor fusion.

Definitions

October 17, 2024 8:17 PM

Proprioception: Sensors that perceive elements of the robots internal state.

Exteroception: Sensors that perceive elements of the state of the external world

Perceptual System: Proprioceptive and exteroceptive sensors combine to make the perceptual system

Sensors: Physical devices that measure physical quantities

Uncertainty: The robots inability to be certain about its internal and external states in order to make a optimal action.

Bit: a basic piece of information is on or off

Pixel: A basic element of image on a camera lens.

Retina: that part of the eye that sees and passes information to the brain

Action in the world reconstruction: Secondary information required for complex sensors that tells information about what the world was like when a reading was taken.

Signal-to-Symbol: Process of making an intelligence response from a sensor

Sensor preprocessing: Sensor signals that are processed in order to extract information.

Electronics: Measuring voltages through switches in a circuit is called electronics

Signal Preprocessing: Process of separating signal information from background noise

Computation: Extracting information using signal processing, then comparing against known information to identify information

Action-oriented perception: use the knowledge about a task to look for stimuli and respond.

Expectation-based perception: use knowledge of environment to guide and constrain sensor data

Task-driven attention: Direct perception is where the robot moves to the direction of information

Perceptual classes: World partitioned into categories, meaning information can be more manageable

Sensor fusion: combining multiple sensors to get better information

Food for Thought

October 17, 2024 8:17 PM

Uncertainty is not much of a problem in computer simulations, which is why simulated robots are not very close to the real, physical ones. Can you figure out why?

Computer simulations are environments that are entirely generated within a computer. If that same robot is simulated inside that environment, then there is no information that the robot is not aware of or how to interpret. Uncertainty is defined as a robots inability to be certain about the state of itself and its environment. Meaning, unless there is a disconnect between the robot, or the simulated environment was programmed specifically to introduce meaningless information, the robot will not experience uncertainty.
Ways to introduce uncertainty into a simulated environment could be introducing sensor noise. Take for example if a simulated robot monitors the temperature of a processor, and if it gets too hot, it starts waving a fan. The simulated environment could be programmed to return garbage readings from the thermometer, such as negative values. In that case, the robot wouldn't be certain on if it should wave the fan or not.

Some robotics engineers have argued that sensors are the main limiting factor in robot intelligence: if only we had more, smaller, and better sensors, we could have all kinds of amazing robots. Do you believe that is all that's missing? (Hint: If that were so, wouldn't this book be much thinner?)

I believe that smaller, better sensors could absolutely lead to more amazing robots. As described in detail in the past two chapters, we know that sensors are electronic devices that report physical quantitative information. Robots are require perception to operate, and perception require: sensors (power and electronics) computation (power and electronics) and connectors. If better sensors were available, smaller, lower powered, lowered processing intensive, then those savings could be passed into computation.
Biological sensors are an excellent example of this. Consider a mosquito; it has no actual brain, just a distributed sensory system. The mosquitos specialized sensors are miniscule, consume a fraction of the insects energy, and allow the mosquito to instantly react to stimuli like carbon dioxide, or the sound of a pulse.

Being able to sense the self, being self-aware, is the foundation for consciousness. Scientists today still argue about what animals are conscious, and how that relates to their intelligence, because consciousness is a necessary part of higher intelligence of the kind people have. What do you think will it take to get robots to be self-aware and highly intelligent? And if some day they are both, what will their intelligence be like, similar to ours or completely different?

I believe that robots will be self-aware and highly intelligent only after are able to self replicate and evolve. By this I mean that a robot will have the ability to analyze its own internal state, and capabilities, then identify any uncertainty conditions or limitations with this own capabilities and identify an adjustment, alteration or reprogramming to solve this problem. This means true machine learning, problem solving, and hypothesizing. Once this feat has accomplished, then a robots capabilities would nearly be limitless.

Unit 4 - Chapter 8

October 17, 2024 8:17 PM

The chapter starts by revisiting active and passive sensors. It describes how either can be simple or complex. Active sensors provides its own stimulus which it perceives the results of. Passive just read data around it.
Other types of sensors are switches, like contact sensors, limit sensors or shaft encoders.
Some switches have been emulated in real life, like a cats whisker.

Another types of switch is a light sensor which measure the amount of light impacting a photocell. Photocells are sensitive to light and that is reflected in the resistance in the circuit they are attached to. They are simple but can see much more than human eyes, including infrared and ultraviolet. Through clever positioning, and usage, lights can be used to detect variability, intensity and breaks in light. Light sensors can also detect polarization. Where filters can be applied to detect for lights of a particular light wave orientation.

Reflective optosensor consist of an emitter and a detector. Usually a light emitting diode and photodiode / phototransistor. Photodiodes are more popular choice for robotics because they are faster.
Reflectance sensors are side by side and detect light of reflecting objects
Break-beam are where light face one another
They can be used to detect distance or presence of objects, features of the object can be detected, if it’s a bar code it can be read, and rotations of a shaft.
No sensor is perfect, reflective can read data from other light sources, such as ambient light.
Calibration can assist with improving performance from elements like ambient light. Visible light is light humans can see, IR light is not visible. IR is usually only used in active sensing, however some specialized use of IR can be such as military night vision. IR is also used for modulation and demodulation of light, used for TV remotes
Break beam sensors detect breaks in light beam.
Shaft encoders detect rotation of a shaft or axle. Marks are put on a wheel, and the sensor detects these marks. Ie: Timing mark on a fly wheel and my timing light.
Using two shaft encoders in perpendicular form is how old mice worked, quadratic shaft encoding - that’s pretty cool.
This similar encoding is used in robot arms as well as cartesian robots (ie 3d printer plotters)
Resistive position sensors also exist, called bend sensors. Not very reliable or used.
Potentiometers, known as pots. Are resistive sensors and the poisiton of the knob controls resistance.

Definitions

October 17, 2024 8:17 PM

Passive Sensor: measure physical property from the environment

Active Sensor: provide their own stimulus and use the interaction of that signal with environment.

Emitter: produces a signal for the detector to perceive

Contact Sensor: detect when a sensor makes contact with another

Limit sensor : when a mechanism has moved to the end of its range

Shaft encoder: detects how many times a motor shaft turns by having a switch click every time the shaft turns

Reflectance sensors are side by side and detect light of reflecting objects

Break-beam are where light face one another

Ambient light: Light from an existing environment

Calibration: process of adjusting a mechanism to maximize its performance

Visible Light: light in the frequency band of electromagnetic spectrum that human eyes can perceive.

Modulated light: light is modulated by turning the emitter on and off, pulsing it.

Demodulator is a mechanism tuned to that frequency and detects the flashes to determine its frequency and decode it.

Quadratic shaft encoding: two shaft encoders are aligned to 90 degrees out of phase. Comparing outputs provides directional changes.

Cartesian robots: Robots of high precision that move arms back and forth along a axis or gear.

Digital: devices that operated or are controlled by discrete values.

Food for Thought

October 17, 2024 8:17 PM

Why might you prefer a passive to an active sensor?
Passive sensors are typically less expensive, both in terms of resources to operate but also to manufacture. Passive sensors are also helpful because reactions can be directly linked to their output, without any additional processing. An example of this would be the Braitenberg's vehicles. Outputs of the passive sensors were directly tied to motor controller input. The result was a perceptive behaviour similar to a living creature.

Are potentiometers active or passive sensors?

Potentiometers are passive sensors, because they react directly to their environment, without providing their own stimuli. They are dependant upon the input voltage, they do not generate or send that voltage.

Our stomach muscles have stretch receptors, which let our brains know how stretched our stomach are, and keep us from eating endlessly. What robot sensors would you say are most similar to such stretch receptors? Are they similar in form (mechanism of how they detect) or function (what they detect)? Why might stretch receptors be useful to robots, even without stomachs and eating

Resistive position sensors are the closest robotic sensor to stretch sensors in a robot. Resistance of a material can be measured which is how these sensors work. These type of sensors were intitially introduced while developing the grandfather of the Nintendo PowerGlove. The device proved somewhat functional, and Nintendo invested, however changed the mechanism of a bend sensor to the current implementation of a sonic measurement. The glove was a horrible failure, with unreliable inputs, and limited game implementation.

Unit 4 - Chapter 9

October 17, 2024 8:17 PM

In this chapter we will learn about much more complicated sensors that will require understand more complex processing but also understanding the underlying functionality of the sensors. These sensors may include ultra sound, laser, vision, GPS etc.

Ultrasonic or Sonar

The first sensor type is ultrasonic or sonar sensing. Ultrasound (from beyond sound) relies on sound frequencies beyond human hearing. It's called sonar, **so**und **na**vigation and **ra**nging.
Sonar is done by echolocation where a sound is bounces of objects and echoes used to find location. Bats and dolphins are then used as examples of echolocation; ie: bats seeing in the dark.

Synthetic (non-biologic) sensors work on a similar behaviour called "time-of-flight" where the amount of time something takes to travel (ie: sound) An emitter sends out a chirp, and measure time to return, then divides by two (Time there and time back) The computation is easy and requires knowledge of speed of sound. The power required for a sonar is quite high compared to processor. Good practice to separate processor power from device power. Sonars have limits on range (degree and distance)

The chapter then goes into investigating sonar uses in other applications. Initially introducing sonar for use in ultrasounds, and medical industry. These function on the properties of sound reflection through water, which is a great prelude in to the next subject of sonar use in submarines. The dangers of sonar are also investigated, including those submarine sonars causing harm to marine animals such as whales and dolphins. Sonar is also used in medical treatments, such as to break up kidney stones.

The next problem with sonar is specular reflection, where sound is bounced off of objects in an environment. This can lead to inconsistent behaviours due to types of materials or surfaces in the room. Another problem is if the reflection never returns. Environments can be altered to reduce this occurrence, but that's not ideal. Instead multiple sensors can work together, past histories or data can be kept and compared.

Laser Sensing

Lasers are excellent at achieving similar function as sonars, with the exception of the laser. Lasers (visible or not) are not always desired for all situations. The time-of-flight measurement still applies, however not appropriate for short distances, as the speed of light can't be measured reliably enough. In these situations, phase shift is used rather than time-of-flight. Lasers are faster and smaller areas of measurement. This allows more measurements at higher resolution. Distance measurement lasers are expensive and large. A visible grid is an interesting way to detect a predictable grid in an environment, if the grid is altered, then objects are detected.

Visual Sensing

Cameras are visual sensors that act like eyes for seeing things. Machine vision is the process of dealing with vision in machines, however robots have particular needs. Understanding visual data from a camera is extremely complex and resource intensive. Fortunately, not a problem for robots typically.

Cameras

Cameras are biomimetic. They capture light from objects in the environment, through an iris, and onto an image plane. That image plane goes through early vision to extract basic information, then high-level vision. Instead of rods and cones in an eye, cameras use charge-coupled devices (CCD). Machine vision then takes over to make sense out of the information. Lens' can focus on a particular item, and affect the depth of field. Image planes are split into pixels. Breaking down an image into a small image is called surface patch. Edge detection is often done.

Model Based Vision

Model based vision is using previous information within the robot to better identify objects within an environment. It involves edge finding and memory services. The objects are identified as models. When an model is identified its approximate size and shape are compared against memory banks. Face recognition is a form of this, except for those who suffer prosopagnosia, or inability to recognize faces.

Motion Vision

Normally motion vision is difficult, but using motion vision, or multiple images and comparing them, static objects can be easily excluded, what's left behind is movement. If any other objects are moving in the environment however, it can become very problematic.

Stereo Vision

Having two eyes means you have binocular vision. Stereo vision (just like motion) involves analyzing two images (one from each ey). The brain is trained to position the two images, how the camera (eyes) are position, as well as how the reconstruction occurs. It provide depth perception. 3d glasses takes advantage of this behaviour by projecting a unique image on each eye, giving the impression of depth.

Texture, Shading, Contours

Analyzing textures, shading and contours of an image can give important clues as to the objects within the environment.

Biological Vision

Biological model based vision is much different than machine vision, and barely understood. Biological vision handles motion vision much better too. Vestibular Ocular Reflex is the act of focusing on an object while either yourself or the object are moving. It's extremely difficult to reproduce, and has been studied heavily. Humans are sensitive to motion, event in periphery. Stereo vision is extremely important as well.

Vision for Robots

Robot vision needs to inform the robot about important things, if its about to fall, object to avoid, finished job etc. Vision processing is extremely complex and responding quickly is difficult. Techniques for simplifying are important such as : use colour, use colour and motion, using small image plane, combining other simpler faster sensors with image data, use previous knowledge about the environment.

It summarizes with an example of self-driving vehicle, and the difficulties surrounding that, including other moving vehicles, no time for slow processing, difficulty in urban environments.

Definition

October 17, 2024 8:17 PM

Ultrasound: means 'beyond sound' refers to a range of frequencies of sound that are beyond human hearing.

Sonar: Sound navigation and ranging

Echolocation: Sound bounces off objects and forms echoes that are used to find one's place in the environment.

Time-of-Flight: they measure the time it takes something (sound) to travel. Ie: sonar

Speed of Sound: a constant that varies only slight due to ambient temperature. At room temp. sound travels at 1.12 feet per millisecond.

Transducer: is a device that transforms one form of energy into another.

Hertz (Hz): A measurement of frequency, a hertz is one per second.

Specular Reflection: mirror like reflection of waves, such as light, from a surface

Discontinuity; discontinuities are sudden and large changes in an environments features.

Laser: Highly amplified and coherent radiation at one or more frequencies.

Resolution: refers to the process of separating or breaking something into its constituent parts.

Machine vision: is the research field that deals with vision in machines, including robots, although Robots have particular perceptual needs related to their tasks.

Biomimetic: they imitate biology, ie cameras work somewhat the way eyes do.

Scene: a collection of objects in an environment

Iris a simple pinhole but usually a lens that allows light into vision system

Image plane: the image plane is the plane to which the image is projected.

Photosensitive: elements that are light-sensitive

Early vision: The first stage of visual image processing occurs with early vision nerves.

Lens: Invertebrates and cameras have lenses, which allow more light to get in, but at the cost of focus.

Focus: objects a certain range of distance from the lens

Depth of field: is the ranges of distances of a camera

Pixels: an equal division of an image plane, typically arranged in a rectangular grid.

Image: The projection of the scene on the image plane is called the image

Surface patch: A small close up part of a larger image

Diffuse reflection: consists of light that penetrates into an object, is absorbed then it comes back out.

Time series: A series of images captured over time

Frame: an individual snapshot in time

Frame grabber: a device that captures a single frame from a cameras analog video signal

Digital image: A digitized image of a analog snapshot

Image processing: techniques used to analyze and manipulate image information captured by cameras or sensors

Edge detection is the first step of image processing and is done by identifying sharp changes of intensity (brightness) in an image

Edge: a curve in the image plane across which there is a significant change in brightness.

Smoothing: a mathematical procedure call convolution, which finds and eliminates the isolated peaks.

Segmentation: process of dividing or organizing the image into parts that correspond to continuous objects.

Model-based values: Using models stored in memory from previous data collection, those models are used to identify objects within image

Prosopagnosia: a neurological disorder that makes it difficult for some people to perform facial recognition.

Human-robot interaction: a field of study that desires to fully understanding, designing and evaluating interactions between humans and robots.

Motion vision: a set of machine vision approaches that use motion to facilitate image processing

Binocular vision: Creatures that have two eyes have binocular vision, this provides stereo vision

Stereo vision: formally called binocular stereopsis, is the ability to use the combined points of view from the two eyes or cameras to reconstruct a 3 dimensional object to perceive depth.

Vestibular Ocular Reflex: (VOR) in which your eyes stay fixed even though your head is moving, in order to stabilize the images

Blob tracking: computers recognize color and movement to track objects without having to actual recognize objects.

Food for Thought

October 17, 2024 8:17 PM

What is the speed of sound in metric units?
According to the textbook, the speed of sound is 1.12 feet per millisecond. We can convert feet to centimetres by multiple by a factor of 30.48. The speed of sound in metric is 34.14 centimeters per milliseconds.

How much greater is the speed of light than the speed of sound? What does this tell you about sensors that use one or the other?
The speed of light is almost 300,000 kilometers per second, compared to the pathetic 331.5 meters per second, we can see that light is immensely faster than sound.
This means that light sensors will receive data much faster than sound, especially when it comes to active sensors. It also means that if light sensors can receive data faster, and if sensors are capable of processing data as fast as it arrives, then the light sensor can read multiple times per each sound sensor. That being said processors that work at the speed of light aren't practical, so processing will be the limiting factor.
If ambient light or background noise is no issue, and uncertainty shouldn't be considered, then light sensors would be the better choice. With one exception: measuring shorter distances. The light travels so fast, that the time-of-flight measurement is too short to be measured accurately.

What happens when multiple robots need to work together and all have sonar sensors? How might you deal with their sensor interference? In Chapter 20 we will learn about coordinating teams of robots.
If multiple robots work together in a close area and all rely on sonars, then the possibility of uncertainty is extremely high. Sonars are active sensors, where a sound wave is transmitted into an environment, reflection occurs, and the receiver reads the sound wave. If multiple robots transmit the same waves, then those waves could be picked up by other robots, causing considerably confusion.
I can think of two ways to mitigate this problem: modification of frequency, and timing of sonar pings. Having the robots use the sonars out of phase. Much like how polarization can be used with light waves to differentiate sources, we could tune individual tuners to individual frequencies such that they do not collide within the same environment. Low to High frequency ranges are from below 1000 Hz to over 500 KHz, that's a huge spectrum.
Another option, if tunable sonars aren't available, then the robots could coordinate between each other and only send individual pings. Communications and pings between robots could occur in a fraction of a section, meaning that the robots would be giving up some granularity or precision of reading, it could still be sufficient. Perhaps, slowing the movements or reactions to sonar input could also assist.

Besides using time-of-flight, the other way to use sonars is to employ the Doppler shift. This involves examining the shift in frequency between the sent and reflected sound waves. By examining this shift, one can very accurately estimate the velocity of an object. In medical applications, sonars are used in this way to measure blood flow, among other things. Why don't we use this in robotics?

Doppler shift sonar involves several aspects that would not be favourable for robotics. Firs t is that medical sonar imagine required much more complex postprocessing, and the image may not be in real-time. Second, doppler shift relies on sound travelling well through water and that's an important aspect for medical imaging. Using doppler in an uncontrolled environment may provide inaccurate results. Finally, time-of-flight sonars are extremely inexpensive compared to doppler.

Since two eyes are much better than one, are three eyes much better, or even any better, than two?

Three eyes could be better than two for many reasons, but it'd could also be a major problem as well. A third eye could provide extra visual information about a given environment, or perhaps even provide information to an angle not previously visible by humans. A third eye could also provide redundancy for humans, as vision problems plague so many people. Unfortunately though, a third eye would also introduce 50% more visual information for the brain to process. "a very large portion of the human brain is dedicated to visual processing" If we increase the requirements on the brain by 50% surely, other processes will suffer.
This led me to some independent research. It makes sense that our brain works so hard to understand its environment, how does a fly with barely any brain handle that same action with dozens of eyes? Well it turns out, my question isn't quite right. Insects, such as the fly, are not capable of understanding (or reconstructing like a robot) their environment. Their eyes are much more capable of processing large amounts of information at a very superficial level. Much like "Blob Tracking" a fly is able to quickly track less distinct images, but they're able to do it extremely quickly. An many insects don't have dozens of eyes, they have dozens of lens' per eye! That means they're able to capture less distinct images from a larger field of view. So an insect doesn't care what's chasing it, it just knows something is.

Forum: Integrated Development Environment Options? (part 1)

October 14, 2024 2:47 PM

Background:
While preparing for this course I noticed various alternative options for Arduino IDE and I'm curious if anyone has tried any of them?

Some of the options I found were:
- Arduino IDE for VisualStudio
- Arduino Pro IDE,
- VS Code Marketplace Enhancement
- Emacs
- PlatformIO
- Etc - leave a comment if I missed one!

I've been developing professionally Microsoft's IDE for a very long. The first version I used, was a barely functional Visual Studio 6.0, and have used practically every version since. I haven't used VS Code very much, as I'm usually in Pro 2024, so I'm really interested in doing a review of the Arduino IDE for Microsoft Visual Studio 2024.

Some criteria I'm going to look into:
- Ease / comfort of use
  - Easy to setup? User friendly? Are there keyboard shortcuts? Is it documented well?
- Available features
  - Intellisense, Source Control, Profilers
- RedBoard integration
  - Can I use it for this course?

Preface:
Git integration

> My primary reason for this review is my desire to use Git. I find it useful for protection codes, especially while developing for more complex systems, as well as provide an easy way to share my work with instructors.
> I was hoping for an all in one solution with Arduino IDE but none was available. Instead I've opted to use Git Desktop and folder based integration - it works fine, but takes too many steps. Sometimes I jump into Command Prompt and use the CLI, but not very often.

Installation
Installation was very easy, and free! I used the Windows Marketplace to install VS Code, took almost no time. Inside VS Code Extensions, there were dozens of Arduino options available! Many options were available with a range of features. Strangely enough, the most downloaded option "Arduino" by "moozyk" seemed to be the worst option, and lowest rated. I opted for "Arduino Community Edition" as it was very well rated, and came complete with Serial port monitor, intellisense, syntax highlighting, board and library manager. Very impressive!

Arduino Community Edition for VS Code - First Impressions

Opening an existing project:
I started easy, with my first challenge - the blinking light. I opened it by selecting the folder. Interestingly enough, VS Code immediately prompted me with an intimidating window, asking if I trusted the author. I was not expecting that, luckily, I am the author so I said "yes". Voila the ino file was opened, with beautiful colour syntaxing. A little less monotone as Arduino IDE, so far so good. Uhoh, another warning, this time I had to install c++ extensions - no problem, VS Code did all the work. A minute later, and I'm back to my ino file - with 13 problems!

It's at this point, that I'm thinking to myself, I haven't specified what I'm coding for, there must be missing libraries and configurations at this point.

And it's at that point that everything came to a halt.

Dead in the Water:
The documentation I found said I'd have to install libraries, they were missing. I had to configure the board, but that touted 'Board Manager' wouldn't open. Similarly, the 'Board Library' failed to open as well. No errors messages, just nothing happened.

It is at this point that I learned the project is dead, and no longer maintained.
I wish I'd found this earlier, but a large thread on the Arduino site concludes this extension for VS CODE is a no go…

There is hope! A community driven group has resurrected a previous fork of the extension, and alive here:
https://github.com/vscode-arduino/vscode-arduino/issues/13
Maybe I should take a look at it after this course!

For now, I'm abandoning Arduino extension to try out **PlatformIO**! Stay tuned, that's coming next…