¡¡¡¡¡¡¡ HEAD Swarming
======= Maarten
¿¿¿¿¿¿¿ feature/WMN

Research document

M. van Wilgenburg, W. Mukhtar, E. van Splunter, M. Siekerman, T. Zaal and M.

Visser

List of Figures

List of Tables

Contents

1 Protocol

1.1 Hardware

For the internal communication between the different systems there are multiple protocols to chose from to transmit data. every subsystem with a microcontroller should be able to communicate with the rest of the system. Because every system should be able to share their information it is necessary to have a multimaster system which allows every system to share their information without being dependent on a single master. To see which protocols are viable multiple protocols will be compared.

Building a modular robot consisting of subsystems, there must be a method to let these "*subsystems*" communicate with each other to exchange relevant information.

Every subsystem features a micro-controller which must be capable of transmitting and receiving data.

1.1.1 TWI

The name TWI stands for "Two wire interface" which strongly resembles Phillps' protocol $I^{(}2)C$. Like the name suggests the bus is build up using two wires, one is used for the clock signal and the other is used for the data signal. Each of these wires is connected to the powerline through a pull-up resistor so every subsystem connected to the line can pull the line down which other systems can see. TWI can be used as a multimaster system which means every subsystem can send it's data without being dependend on a single master.
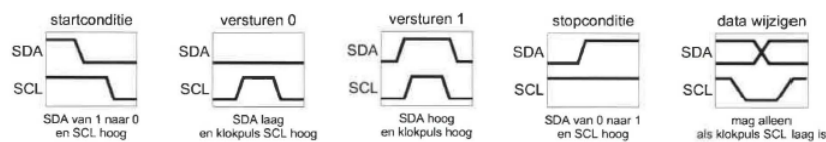

Figure 1: Data conditions

In the above figure multiple data and clock combinations are shown, these combinations define the message start, sending a zero, a one and a stopbit. The system that wants to send information starts a message bij changing the datasignal from one to zero on a high clocksignal. All systems that are listening now know one of the systems is going to send a message. Data can be send by changing the dataline from zero to one or the other way around during a low clocksignal, the other systems will read this bit when the clocksignal becomes high again. Once the system is done sending its message it will let the other systems know by sending a stopbit.

Figure 2: Structure of a TWI message

In the above picture the structure of a message is shown. As explained before the message starts with a startbit after which a small header follows. The header contains information about who is going to be adressed by the message and if you want to read or write. After this the adressed system will respond with an acknowladge to make sure the information has come across. When the acknowladge is recieved the system will start sending data, followed by and acknowladge from the reciever after every recieved byte of data. when the system is done sending all of its data and recieved the last acknowladge it will send the stopbit which means all communication has ended.

TWI has a few benefits, because it is a simple serial protocol it only uses two wires. this saves a lot of space in the hardware design. Also, it has the possibility to be used as a multimaster system. The speed of the communication depend on the frequency of the controller that will be used.

1.1.2 SPI

SPI or Serial Peripheral Interface is, like it names suggests, a serial interface. This interface can be connected in two ways, each having its own advantage. How this works will be explained using the following figure.
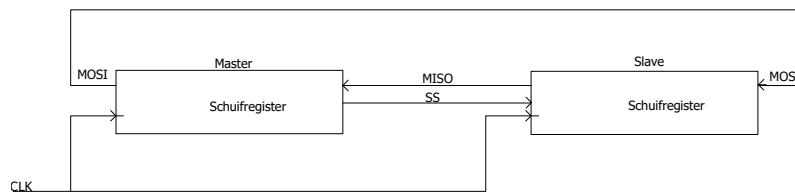

Figure 3: SPI connections

SPI works by selecting its slaves using its slave select. Slave select is a single bit used to define which slave the master will be talking to. When the master has selected its slave it will be able to shift its data from its shift register into the shift register of the slave using the MOSI output. MOSI mean master out slave in, so if the master asks data from the slave it will be send using the MISO, master in slave out. Knowing this there are two ways to connect the slaves, putting them in series or parallel to eachother.
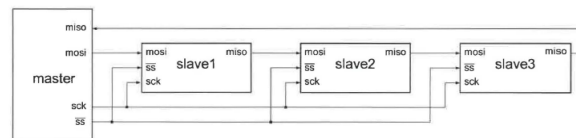

Figure 4: SPI connected in series

When put in series only one slave select will be used to select all the slaves. Each slave will send its data to the next slave through the MISO line. The disadvantage of this method is when you want to address the last slave in line it has togo through all the slaves, meaning it will costs a lot of extra clockticks to

put it through. Another disadvantage of this method is you can not add extra slaves to an excisting series of slaves because the path needs to be closed else the slaves will not be able to send their data to the master.
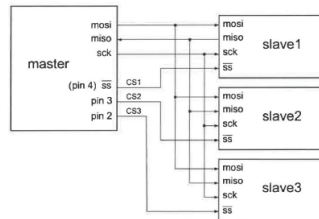

Figure 5: SPI connected parallel

The second way to connect the slaves is parallel, this means the master can address all the slaves individually, which will make it faster. The downside of this is you will need a different slave select for every slave. Which means this takes up more outputs from the controller and more space on the circuit boards. Another downside of SPI is that it can not be used as a multimaster system this downside means it is not viable for the project.

1.1.3 U(S)ART

The third serial communcation typically used on microcontrollers is U(S)ART. This protocol has two variaties, the universal asynchronous reciever/transmitter and the synchronous. The difference between the two is that the synchronous uses an extra line to send a clock signal. In asynchronous mode the devices synchronice with a startbit at the beginning of the message. Both of the protocols have a TX and a RX line, the transmit and recieve line. Like the name says these are the lines to transcieve the data. Each device connected has two registers, one for recieving data and one for sending. These registers are eight to ten bits long, eight or nine data bits and one or no parity bit. The parity bit can be used as a check to see if the data was recieved correctly by checking if the amount of ones is even or odd. The message is ended with one or two stop bits. So a typical U(S)ART frame looks like the figure below.
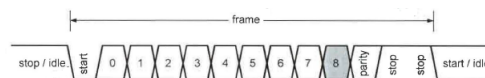

Figure 6: Typical U(S)ART frame

Depending on the software protocol used it is possible to have a multimaster system using one U(S)ART connection on every device. Every message should get a header in which the addressed divece is selected.

1.2 Software

2 Relative Distance

Building a swarm robots consisting of multiple units, location awareness may be required to ensure an more efficient operation. Implementing location awareness into a swarm of robots could result in better cooperation. For example: exploration based tasks can be executed on a much lower time scale. With location awareness the swarm can spread out evenly making sure that every part of the perimeter will be explored thoroughly.

With location awareness it is also possible to divide the swarm into multiple groups. Because the location of every individual is available these individual groups can be formed very quickly by selecting the nearest unit. This also enables quick assists when an individual robot gets stuck or needs to execute a task that require more than one unit.

Relative distance can be measured in various ways using various methods. The most common methods are *"Received Signal Strength Indication"* and *"Time Of Flight"* abbreviated as RSSI and TOF.

2.1 Received Signal Strength

3 Onderzoek

Hier tekst plaatsen

3.1 Propulsion,Actuators and Effectors

In this chapter the sub-questions about the propulsion, actuators and effectors are researched and answered. First of all, the possible types of propulsion will be researched. Then the various kinds of actuators with a suitable effector will be considered. Depending on the situation different types of propulsion is used. For example a robot that operates in an environment with water would not be implemented with DC motor drivers and tyres. The robot build in this research should be able to operate on Mars in the future. Therefore the robot cant get stuck easily since that will cost millions, since the robot cannot complete his mission. Since swarming and not propulsion is the main goal of this research a simple way of locomotion would suffice. However there should be the possibility to implement a different kind of propulsion in the future, so that the robot can be used on Mars.

3.1.1 Effectors

The way a robot moves is determined by its effector with his corresponding actuator. Possible types of effectors are:

- Wheels

- Leg(s)

- Catepillar tracks

- Air cushion

- Propeller

- (Superconductor)

Each of these effectors have pro's and con's in different situations. Tyres are easily implemented, but there are various ways to implement these. The amount of tyres used will affect the handling of the robot. With a minimum two tyres, movement can already be realized. The downside is that driving with two wheels is not stable, only with algorithms that can balance the robot stable movement can be accomplished. It would be easier to implement more than two wheels so that the robot is more stable and movement is easier.

Elke van deze effectoren hebben voor en nadelen in verschillende situaties. Wielen zijn eenvoudig te implementeren, er kunnen verschillende aantallen wielen worden aangebracht. Het aantal wielen heeft wel invloed op het rijgedrag van de robot. Met twee wielen kan er worden gereden, mits de robot in balans wordt gehouden. Dit kan worden gedaan door middel van een zwenkwiel, of door een regelsysteem te bedenken dat de twee wielen de robot in balans houden. Dit laatste kost veel energie en is niet eenvoudig. Met drie wielen staat de robot al een stuk stabieler. Het nadeel hiervan is dat sturen soms moeizaam kan zijn. Met vier wielen is de robot nog stabieler en kan er op verschillende

manieren worden gestuurd. De wielen kunnen tegengesteld worden aangedreven
waardoor de robot draait. Dit kost wel meer energie dan nodig is aangezien er
extra wrijving ontstaat. Een optie is de voorste wielen te laten draaien, waar-
door de robot in de richting van de wielen zal voortbewegen. Zoals bij een auto
het geval is. Dit is weer lastiger te implementeren aangezien er een soort van
stuurmechanisme moet worden bedacht.

3.1.2 Actuatoren

De keuze van de actuator hangt sterk samen met de effectoren. ¡3

3.1.3 Aandrijving

4 Swarm Communication

There are several requirements to realise a robot swarm. Each member of the
population needs to communicate with the rest of the swarm. To give the
members full freedom of movement the communication also needs to be wireless.
It is important that the population size of the swarm is dynamic, because of the
scalability of the swarm. As mentioned earlier a swarm does not have a master.
So each member needs to be capable to maintain the network. These networks
also must be able to reroute around nodes that have been lost.

To create an network with those characteristics a (wireless) mesh network
((W)MN) is a possible solution. A mesh network relies on communication nodes
in which data is distributed. All of the communication nodes cooperate in the
distribution of data in the network. Each node of the network is connected to one
or more nearby in range node(s) and forwards data on behalf of the other nodes.
[?] A example of an mesh network is illustrated in figure ??. To ensure a realiable
network, self-organization and topology control alogorithms are needed.[?] Mesh
networks can use self-healing algorithms such as Shortest Path Bridging (SPB)
to restore and reroute the connection around broken nodes/clients. Shortest
path bridging calculates the most favorable connections to the clients with the
lowest path costs. [?] Other usefull algorithms are listed in [?]. The most
efficient algorithm for path bridging depends on the situation where the clients
are operating in which is described in [?].

A wireless mesh network (WMN) is a dynamically self-organized and config-
ured network. [?] Each node or client in the network is able to create an ad hoc
network to maintain the mesh connectivity, this is called client WMN's. These
networks are able to reroute around nodes that have been lost. On large scale
projects wireless mesh networks are costefficient because of the saving of wired
connections.[?]

There exist several techniques to accomplish this method Examples of pos-
sible protocols:

For close range obstacle avoidance, the robots will need some kind of prox-
imity sensing technique. The sensor needs to differentiate between different
"treats" for the robot to function properly. For example; when the sensor de-
tects a small hill, which the robot can move over it should not trigger the robot
to move away from it. But when it detects a big obstacle it should trigger the
robot to move away. This is important because the robots will eventually be
expected to move across rough terrain (like Mars). In the next paragraphs some

different proximity sensor techniques will be discussed. Price is also a important factor while comparing these techniques. Multiple robots must be made on a tight budget, so costs should be cut where possible.

4.0.1 Ultrasoon sensor

Een veel gebruikte proximity sensor op simpele robots is de ultrasoonsensor. Dit komt omdat deze in vergelijking tot andere sensoren een goedkope oplossing is. Een ultrasoon sensor stuurt een ultrasoon geluid uit dit geluid wordt teruggekaatst tegen het object en vervolgens weer opgevangen door de sensor. De tijd die het geluid er over gedaan heeft wordt gemeten. Met een simpel rekensommetje wordt vervolgens de afstand bepaald. Een probleem met de ultrasoon sensor is dat het moeilijk is om te discrimineren tussen verschillende objecten. Hierdoor wordt het moeilijk om onderscheid te maken tussen bijvoorbeeld een heuveltje (waar de robot wel overheen kan) en een obstakel die de robot moet ontwijken. Door de atmosfeer op mars wordt geluid ernstig gedempt hierdoor is ultrasoon niet implementeerbaar op mars [**?**]. Voor testen op aarde is het echter wel een goed werkend en bewezen techniek.

4.0.2 Lidar sensor

Lidar staat voor "Light detection and ranging". Deze sensor werkt volgens het zelfde principe als de ultrasoon sensor. Het is al bewezen dat de lidar techniek werkt op Mars. Met de Phoenix missie is er een lidar systeem gebruikt om wolken en atmosferische stof te meten[**?**]. Het licht van de laser kon wolken meten op meerdere kilometers hoogte. Omdat licht zich zeer snel verplaatst, moet de elektronica in een lidar sensor ook snel zijn. Hierdoor is de prijs in vergelijking tot de ultrasoon sensor hoog.
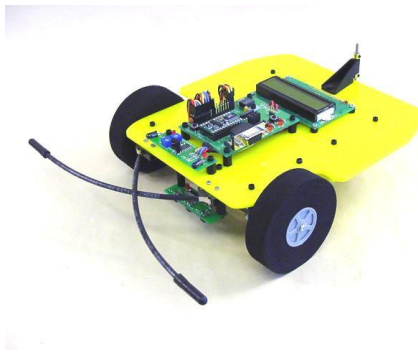

Figure 7: Robot met mechanische proximity sensor

4.0.3 Mechanische sensor

Eén mechanische proximity sensor bestaat meestal uit twee onderdelen. Dit zijn de actuator, dit is meestal een simpele druk schakelaar. En de arm of "bumper", dit is het deel dat het object aanstoot en zorgt dat de mechanishe kracht op de schakelaar wordt uitgeoefend figuur 7 . De kracht van deze techniek is zijn

eenvoudigheid. Er hoeven geen ingewikkelde signalen verwerkt te worden, enkel het aan of uit signaal van de actuator. Een groot nadeel van deze techniek is dat de sensor fysiek tegen een object aan moet komen om het te detecteren. In een zwerm van relatief fragiele robots is dit niet ideaal. Wanneer de robots zich op ruiger terrein bevinden, zal het gebruik van een "bumper" of arm ook niet ideaal zijn, deze kan namelijk vast blijven zitten tegen of achter obstakels.

4.0.4 Sensor choice for the robot

Now the question is which of the different sensor techniques proposed, is the best to implement on the swarm robots. While answering this question the following points will be taken into account:

- The sensor must be easy to reproduce because multiple robots must be made from the swarm.

- The budget is limited, therefore the sensor price should be cheap compared the the total price of one robot.

- Energy usage should be as low as possible to increase the operation time per robot.

- For budget reasons there has been chosen to not take into account if the sensor would work on Mars.

Looking at these points it becomes clear that the lidar sensor wouldn't be the best choice. It doesn't really have any outstanding pros compared to the other two techniques and it's much more expensive. Currently the cheapest lidar sensor would cost about $100e$. The ultrasonic and mechanical sensor wouldn't cost more than a few Euro's. As talked about before the mechanical sensor has a few drawbacks compared to the ultrasonic sensor. For these reasons the ultrasonic sensor seems the best choice. The only issue left to solve is how to sense the difference between obstacles which the robot has to avoid, and things the robot doesn't have to avoid, but still might be detected by the sensor. This will be discussed in the next paragraph.

4.0.5 Obstacle discrimination with an ultrasonic sensor

An ultrasonic sensor is usually used to detect if there is some obstacle in the way. In this case there is made no difference made between a obstacle that the robot can move over, or a obstacle that should be avoided. For example: a small slope or hill might be detected as an obstacle, but in truth the robot can move over it. The robots developed for this program will eventually be able to cross harsh terrain. In this case the proximity sensor will need to discriminate between these situations. Working with ultrasonic waves there are multiple domains to work with. These are time, frequency and amplitude. Time is used to determine the distance between the object and the sensor. The change in frequency and/or amplitude contains information about the shape of the object**??**. In the paper "Object recognition with ultrasonic sensor" is concluded that looking at the amplitude over time contains enough information to discriminate between object**??**. The example is given of an object with separated surfaces of 3,5cm, the echo from the second surface will be delayed by 0,2msec relative to the echo

from the first surface. This delay can be easily detected with a microprocessor. The reason looking at frequency isn't the first choice is because of the higher hardware requirements it would require to properly sample the signal.

5   Bibliografie