International Journal of ASRO
Volume 11, Number 1, pp. 181-191
Jan 2020

DESIGN OF AUTOPILOT TRACKING CONTROL AT A PROTOTYPE TANK BASED ON ARDUINO

Sunarta [1], I Nengah Putra A [2], Cucuk Wahyudianto [3], Muhammad Husni Afandi [4]

[1,3,4]*Indonesia Naval Technology College, STTAL Surabaya Indonesia*
[2]*Indonesia Defense University, Unhan Jakarta Indonesia*

ABSTRACT

The Indonesian Navy is one elements of the Indonesian nation which has the duty to maintain the unity and unity of the Republic Indonesia. The Indonesian Navy requires a lot of equipment, defense equipment and supporting equipment. Unmanned Tank is one vehicle that can be used as a combat vehicle without involving troop personnel in the tank that can be controlled remotely through wireless data communication. The design of the location-based prototype tank autopilot control system is a research in answering for the development of unmanned warfare technology. Prototype unmanned tank using GPS sensor bases,and compass determining the location, orientation and direction of the prototype. The application of the compass as an evaluation of the tank bow course error towards the direction of the destination, while GPS is used as a parameter for determining the location of the destination location. The combination of the two sensors is used as a base information on the control system on autopilot. From the results of tests that have been carried out obtained manual control systems and autopilot in accordance with the itinerary that has been made. The vehicle tank can respond in real time to any location and direction errors, so that corrections can be made to reach the destination correctly. Weather problems become a barrier factor in getting location data from GPS. During the test in sunny conditions obtained control results with a location error factor of less than 5 meters.

Keyword *: Prototype, control, autopilot*

1. INTRODUCTION

The Indonesian Navy is one of the elements of the Indonesian people who have the duty to maintain the unity and integrity of the nation. In carrying out the task of maintaining the sovereignty of the nation, the TNI requires many instruments, defense equipment and supporting devices. One of the supporting defense equipment is supporting defense equipment in carrying out the security duties of the Republic of Indonesia. Supporting defense equipment is a support equipment that is modern and sophisticated in order to overcome difficult problems, especially on the battlefield.

In this era, the development of an increasingly modern era often has an impact that has the potential to present problems that are complicated enough to be solved. National stability and national security are victims of this development. Often the nation's problems can trigger wars between nations and wars between brothers within the country.

Unmanned Tank Alutsita is one vehicle that can be used as a combat vehicle without involving troops in the alutsita tank in an auopillot control via telemetry data communication. The ability to roam unmanned tanks with the autopilot system allows unmanned tanks to move to the points of combat location as specified. Utilization of unarmed military equipment on the battlefield is the right solution to secure the sovereignty of the Republic of Indonesia. In realizing the sovereignty of the Republic of Indonesia, the author tries to raise it as a final project theme with the title of the final project Design of Autopilot Tracking Control at A Prototype Tank Based on Arduino, the authors focus on the design of the autopilot control system on the armaments system.

In this research, it is necessary to formulate the problems in the Design of Autopilot Tracking Control at A Prototype Tank Based on Arduino. Some of these problems are :

a. What is the autopilot control system on a prototype tank vehicle?

b. What is the manual control system on the prototype tank vehicle?

c. How do I access the sensors needed on a tank prototype vehicle?

The purpose of this design build research is :

a. Autopilot control system on the Indonesian Armed Forces Alutsista Tank.

b. As the application of a system of tank defense equipment that does not involve personnel in it.

c. As a solution to the problem of unarmed tank defense equipment for the Navy.

2. RESEARCH METHOD

2.1 Research design

Design of Autopilot Tracking Control at A Prototype Tank Based on Arduino is a design research, to translate the analysis results of a system into a programming language to describe in detail the components of the system being implemented. While the building / system development is the activity of creating a new system or replacing or repairing an existing system either in whole or in part. Thus, the design is an activity of translating the results of the analysis into software which then creates or improves an existing system.

In the final project, Design of Autopilot Tracking Control at A Prototype Tank Based on Arduino applies land media as the media for tank vehicle operation. The control system used applies a manual control and autopilot control system. In manual control work systems, tank motion is determined based on instructions or commands from the remote, while autonomous work systems implement a programmed auto motion system. Autonomous work systems work based on the following conditions, viz:

a. Auto pilot motion commands

The tank will make a flight or trip based on points that have been determined on the ground station control.

b. Low battery

When the tank is on a travel mission, the Arduino unit analyzes the battery usage, battery level and estimated remaining battery. If the battery is in a weak condition the system will give a warning of a low battery and will carry out an autonomous movement back to the home base

2.2 Electronic Systems Analysis

In the Design of Autopilot Tracking Control at A Prototype Tank Based on Arduino, the main unit device uses Arduino Mega 2560. In its application Arduino Mega will function as follows :

a. Processing telemetry communica-tion data via serial port.

b. Running the function of managing the use of battery resources, perform operating time estimates

c. Running a warning system, providing information and warnings to the Home Base regarding battery level.

d. Processing GPS sensor data, Accelerometer, Gyro, Compass, battery current and battery voltage.

e. Run the manual control function based on commands from the remote.

f. Perform auto pilot control functions based on planned location points.

Perform all location monitoring functions, tank orientation on yaw axis, picth, roll, battery level and estimation of tank operation in real time.

2.2.1 System Planning

In this research, the Design of Autopilot Tracking Control at A Prototype Tank Based on Arduino, the planned system is expected to be able to perform all the functions and features described in the previous sub-chapter.

2.2.2 Electronic System Design

The design of electronic systems in tanks takes into account the needs needed in tanks. These needs in general must be able to be met by electronic devices, so that the types and capabilities of electronic devices must be appropriate and appropriate. The minimum device capability limit must be in accordance with the needs and the device capability is strived above the level of need. This is necessary so that the device does not work continuously at the maximum capability limit that is at risk of system failure or system failure. Thus the ability of the device above the requirement is one of the methods chosen to resolve resilience and failure solutions.

In general, the work systems and processes in electronic tanks can be explained in the following figure.

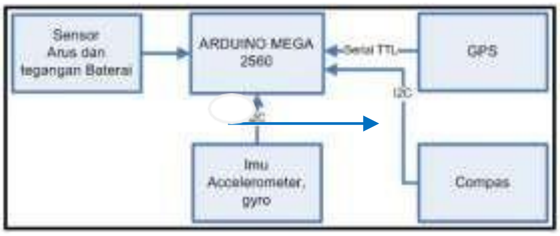


Fig.1 Tank Electronic Device Block Diagram

In the image above can be explained the working parts and functions of the electronic device, as follows :

a. Arduino Mega 2560

Arduino mega 2560 microcontroller device as the main process unit and contains the program code stored in flash memory.

b. Compas sensor

The HMC5883L compass sensor works to get the vehicle's orientation angle to the ground.

1) IMU sensor MPU9250

An inertia sensor that functions to measure the inertia angle of the tank (yaw, pitch, roll), the IMU consists of 3 sensors namely ITG- 3200 MEMS triple-axis gyro, ADXL345 (triple-axis accelerometer) and HMC5883L (triple-axis magnetometer)

2) GPS sensor

GPS is a sensor to read the position of the location of GPS devices, GPS devices used in this study utilize GPS devices on Android devices and the GPS module NEO-M8N which is installed on the main unit.

3) Battery level sensor

A sensor that functions to measure battery level and battery power usage. The module used is APM PIXHAWK Ardupilot Power Module. The condition of the battery measured is the voltage and current of the battery when it is in standby or when it is operating. This sensor is used to estimate tanks when carrying out operational missions.

4) Battery Lippo 3S

As the main power source of the author using the Lippo 3S 3300 mA battery, the Lippo battery was chosen because of its ability to distribute power tens of times over its nominal current size.

2.2.4 System Flow Chart

In the research of Design of Autopilot Tracking Control at A Prototype Tank Based on Arduino flow diagram can be explained in the following Flow Chart at Figure 2:

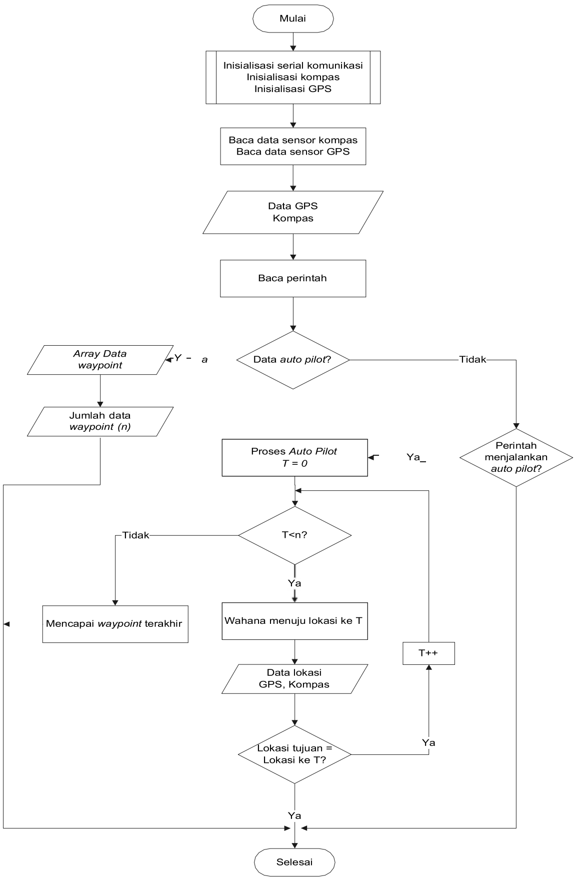


Fig.2 Flowchart

2.2.5 Research Design

Design of Autopilot Tracking Control at A Prototype Tank Based on Arduino by applying the research stages, as follows:

a. Preliminary studies

Preliminary studies are study processes to obtain information about the research to be conducted.

b. Problem analysis

Problem analysis is a study to find out the causes of problems, as well as alternative solutions and later solutions to problem solving.

c. Need analysis

According to experts needs analysis is needs analysis is one of the many critical activities in the process of engineering the need for software to understand the realm of problems of the running system and the realm of solutions of the system to be made (Yen et al., 1998). Requirement analysis is part of the process of software requirements that play a role in bridging the gap that often occurs between the level of engineering needs and software design (Pressman, 2008). Needs analysis aims at perfecting existing needs to ensure stakeholders understand it and find faults, omissions, and lack thereof if any (Wiegers, 2003).

d. Design analysis

Is the process of selecting tools to analyze data, so that the formulation of the problem can be solved and the objectives can be proven. On this stand the writer must choose a data analysis tool in accordance with the data to be obtained in the field study. In this step the author can determine the tools that can help to determine the appropriate tank design and objectives can be realized.

e. Design

Based on a study on the design analysis in the previous paper, information can be obtained to carry out the Auto Pilot Full Build Design Process in the Arduino Mega 2560 Prototype Tank Based Prototype and the development steps.

3. DESIGN, IMPLEMENTATION AND TESTING OF THE SYSTEM

3.1 Design of System

In designing an auto pilot control system on a prototype tank, an analysis is carried out to obtain the appropriate system design. In this design the authors set several sensors for the purpose of tank control on autopilot, namely:

a. GPS, to get locations in latitude and longitude format.

b. Compass, to get the bow of the prototype tank against the angle of the earth.

c. Imu, inertia measurement unit is a sensor to get the orientation of the prototype tank attitude, in the composition of the pitch and roll attitude.

Two sensors namely GPS and compass are the main components in determining the direction of motion of the prototype tank towards the destination location. The two sensors are formed in a control with closed feedback so that the system can correct errors in the position of the prorotype tank in approaching the destination point.

To explain the control in its function can be described in the following closed loop control.

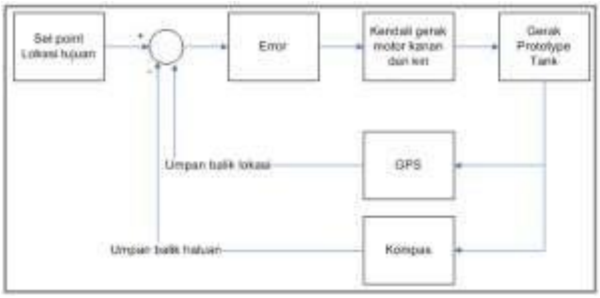


Fig.3 Prototype Full Closed Loop Control From

the drawing of the control design or control of the tank's motion using GPS and compass data feedback, it can be described as follows.

a. Destination location data or on the control system as a set point, is the destination location point or point. The destination location has coordinates in latitude and longitude. In the motion system distance can be obtained assuming that the prototype tank is outside the destination point area. In the system settings, you can do the area settings in meters, namely 1,2,3,4,5,6,7,8,9 and 10 meters. This is done because GPS cannot obtain location accuracy up to 100 percent. So the approach of the location of the prototype tank in the destination point is carried out so that the prototype tank is declared to have reached the destination point. Thus the prototype tank can proceed to the location of the next destination point. The indicator if the prototype tank has reached its destination can be calculated using the following equation, using the haversine formula it can be calculated to get the distance of a circle between two locations

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi 1 \cdot \cos\varphi 2 \cdot \sin^2(\Delta\lambda/2) \ldots\ldots\ldots\ldots\ldots\ldots \quad (4.1)$$

$$c = 2 \cdot \operatorname{atan2}(\sqrt{a}, \sqrt{(1-a)})$$
$$d = R \cdot c$$

Where :

φ is the latitude

λ is longitude,

R is the radius of the earth (radius = 6,371km)

b. Bow data, formula for determining bow direction between two points. The bow angle is used to determine the navigation direction of the prototype tank. The bow can be defined as the direction or angle, between the north-south lines of the earth or meridians and the lines connecting targets and reference points. This means that to achieve certain goals it is necessary to adjust the direction of the position with the bow. The use of a compass as an instrument, can provide direction information for navigation prototype tanks. The formula to get the bearing values between two different latitudes and longitudes can be used the following formula:

$\beta = \operatorname{atan2}(X, Y)$,

Where, X and Y are two quantities (latitude and longitude) and can be calculated :

$X = \cos\theta b * \sin\Delta L$

$Y = \cos\theta a * \sin\theta b - \sin\theta a * \cos\theta b * \cos\Delta L$

3.2 Implementation

For the implementation of the prototype tank autopilot control system, some things that need to be prepared are as follows.

3.2.1 Module GPS

Is a prototype tank location identification module in latitude and longitude format. The prototype tank location data can be updated periodically based on data updates obtained from GPS satellites. By default the GPS data device is updated with a 5hz update frequency which means the GPS is updated 5 times every second.

3.2.2 compass module

Is a module to get the direction of the prototype tank's bow towards the earth, the bow's angle serves to get the correction angle to the target angle based on the calculation of the prototype tank's bow towards the destination location. Simple calculations used by the author to determine the direction of the destination location, written in C ++ program code.

3.3 Testing

In this sub-chapter the author will test the system that has been made, testing is carried out in stages per module module which is integrated in the prototype tank. The following process is the test carried out by the author on the prototype tank.

3.3.1 GPS Module Testing

To do GPS module testing, the steps you need to take are connect the GPS module to the Arduino Mega microcontroller as shown in the following 3.1. In Figure 4.2 the GPS module is connected with Vcc 5V, ground and serial 1 as data communication. In this testing phase, the author displays the GPS reading data on the Arduino serial monitor.

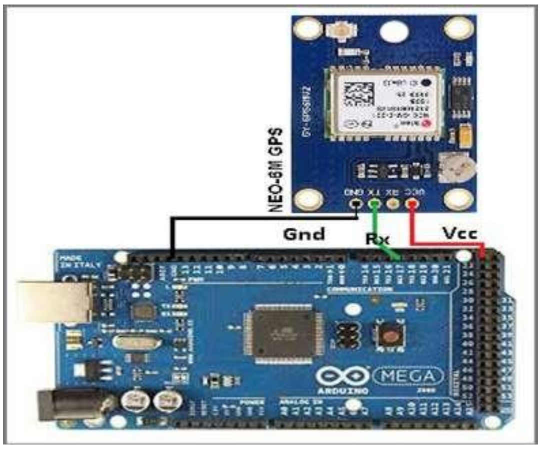
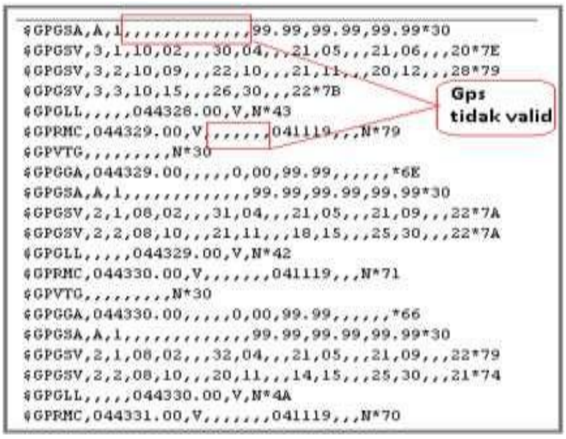


Fig.4 GPS Test Network

$GPRMC,045217.00,A,0726.98874,S,11243.33582,E,0.011,,041119,,,A*67

In the first GPS testing phase, the author will insert the Arduino program code to display the NMEA gps data, which can be seen in Figure 4.

In Figure 5 the test results show an invalid NMEA sentence, this is because the GPS has not received a fixed satellite signal and also because the number of satellites received is less than the minimum value of a 3d fix satellite, which is 4 satellites. 3 satellites to get location in 3 dimensions and 1 satellite to get time data synchronization.

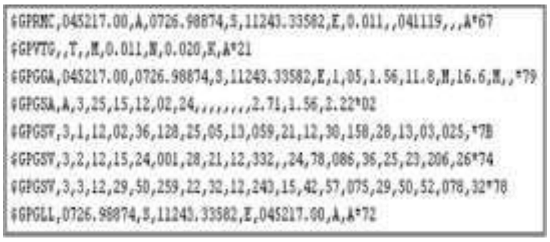


```
$GPGSA,A,1,,,,,,,,,,,,,99.99,99.99,99.99*30
$GPGSV,3,1,10,02,,,30,04,,,21,05,,,21,06,,,20*7E
$GPGSV,3,2,10,09,,,22,10,,,21,11,,,20,12,,,28*79
$GPGSV,3,3,10,15,,,26,30,,,22*7B
$GPGLL,,,,,044328.00,V,N*43
$GPRMC,044329.00,V,,,,,,,041119,,,N*79
$GPVTG,,,,,,,,,N*30
$GPGGA,044329.00,,,,,0,00,99.99,,,,,,*6E
$GPGSA,A,1,,,,,,,,,,,,,99.99,99.99,99.99*30
$GPGSV,2,1,08,02,,,31,04,,,21,05,,,21,09,,,22*7A
$GPGSV,2,2,08,10,,,21,11,,,18,15,,,25,30,,,22*7A
$GPGLL,,,,,044329.00,V,N*42
$GPRMC,044330.00,V,,,,,,,041119,,,N*71
$GPVTG,,,,,,,,,N*30
$GPGGA,044330.00,,,,,0,00,99.99,,,,,,*66
$GPGSA,A,1,,,,,,,,,,,,,99.99,99.99,99.99*30
$GPGSV,2,1,08,02,,,32,04,,,21,05,,,21,09,,,22*79
$GPGSV,2,2,08,10,,,20,11,,,14,15,,,25,30,,,21*74
$GPGLL,,,,,044330.00,V,N*4A
$GPRMC,044331.00,V,,,,,,,041119,,,N*70
```

Fig.5 Invalid NMEA data test results

Each NMEA sentence starts with an identifier like $ GPRMC, to identify the sentence, followed by GPS parameters such as latitude and longitude, separated by commas. The string ends with an asterisk and a two-digit checksum. The most important NMEA sentence used is RMC (Recommended Minimum C), which contains all important navigation parameters: time, latitude, longitude, speed, course, and date. Following is the sentence of the RMC test results after the GPS gets valid data:

From the NMEA sentence, the writer takes the most important data from NMEA information, viz:



```
$GPRMC,045217.00,A,0726.98874,S,11243.33582,E,0.011,,041119,,,A*67
$GPVTG,,T,,M,0.011,N,0.020,K,A*21
$GPGGA,045217.00,0726.98874,S,11243.33582,E,1,05,1.56,11.8,M,16.6,M,,*79
$GPGSA,A,3,25,15,12,02,24,,,,,,,,2.71,1.56,2.22*02
$GPGSV,3,1,12,02,36,128,25,05,13,059,21,12,30,158,28,13,03,025,*7B
$GPGSV,3,2,12,15,24,001,28,21,12,332,,24,78,086,36,25,23,206,26*74
$GPGSV,3,3,12,29,50,259,22,32,12,243,15,42,57,075,29,50,52,078,32*78
$GPGLL,0726.98874,S,11243.33582,E,045217.00,A,A*72
```

Each field can be explained as follows

045217.00 = is 4:52:17 UTC time

A = Status, A = active or V = invalid

0726.98874, S = Latitude 7 ° 26.98874 'S

11243.33582, E = Longitude 112 ° 43.33582 'E

0.011 = Movement speed 0.011 knots

041119 = Date 04-11-2019

A = Mode, A = autonomous, D = differential, E = estimate

* 67 = checksum

Because each character is received from a GPS sentence, it will be checked against the format string fmt [], which is an NMEA RMC (Recommended Minimum) sentence with data replaced with lowercase letters. Each character is tested as follows:

a. If the received character is '$', the status pointer is reset to the beginning of the string fmt [].

b. If the received character matches the next character in the string fmt [], the pair is successful.

c. If the characters in the string fmt [] are lowercase, determine how to process the received character, and the match is successful.

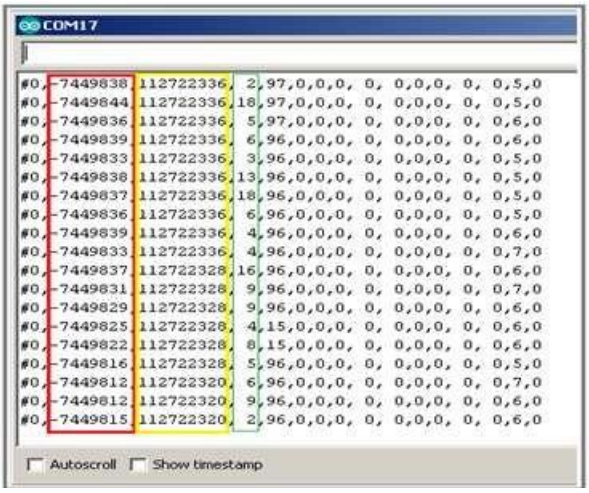
d. Otherwise, a match occurs and the status pointer is reset to the beginning of the string.

e. The letters 'd', 'e', and 'a' read the numbers received and accumulate values in one temp or ltmp variable. For example, a 'd' in the string fmt [] reads a decimal number into the variable temp.

f. The letter 'j' collects numbers until the maximum decimal point is found.

g. The last letter in each field copies the temporary variable to the appropriate GPS variable; for example, the letter 'l' copies ltmp into the latitude variable, Lat.

In the second GPS test, the authors add code to parse NMEA sentences, so that the output can be obtained latitude, longitude and speed data. Figure 4.4 follows is the result of GPS testing that has been carried out by the NMEA parse process to obtain original location data.



```
#0,-7449838,112722336, 2,97,0,0,0, 0, 0,0,0, 0, 0,5,0
#0,-7449844,112722336,18,97,0,0,0, 0, 0,0,0, 0, 0,5,0
#0,-7449836,112722336, 5,97,0,0,0, 0, 0,0,0, 0, 0,6,0
#0,-7449839,112722336, 6,96,0,0,0, 0, 0,0,0, 0, 0,6,0
#0,-7449833,112722336, 3,96,0,0,0, 0, 0,0,0, 0, 0,5,0
#0,-7449838,112722336,13,96,0,0,0, 0, 0,0,0, 0, 0,5,0
#0,-7449837,112722336,18,96,0,0,0, 0, 0,0,0, 0, 0,5,0
#0,-7449836,112722336, 6,96,0,0,0, 0, 0,0,0, 0, 0,5,0
#0,-7449839,112722336, 4,96,0,0,0, 0, 0,0,0, 0, 0,6,0
#0,-7449833,112722336, 4,96,0,0,0, 0, 0,0,0, 0, 0,7,0
#0,-7449837,112722328,16,96,0,0,0, 0, 0,0,0, 0, 0,6,0
#0,-7449831,112722328, 9,96,0,0,0, 0, 0,0,0, 0, 0,7,0
#0,-7449829,112722328, 9,96,0,0,0, 0, 0,0,0, 0, 0,6,0
#0,-7449825,112722328, 4,15,0,0,0, 0, 0,0,0, 0, 0,6,0
#0,-7449822,112722328, 8,15,0,0,0, 0, 0,0,0, 0, 0,6,0
#0,-7449816,112722328, 5,96,0,0,0, 0, 0,0,0, 0, 0,5,0
#0,-7449812,112722320, 6,96,0,0,0, 0, 0,0,0, 0, 0,7,0
#0,-7449812,112722320, 9,96,0,0,0, 0, 0,0,0, 0, 0,6,0
#0,-7449815,112722320, 2,96,0,0,0, 0, 0,0,0, 0, 0,6,0
```

Fig.6 Valid NMEA test results

a. From the test results of Figure 6, it can be explained as follows:

b. The red color shows the latitude data

c. Yellow indicates longitude data

d. Green indicates speed data

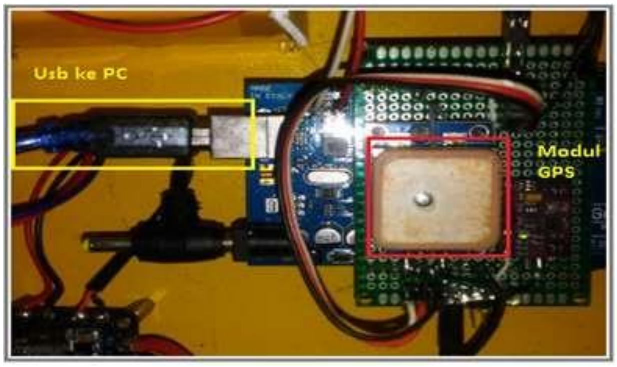


Fig.7 The GPS module

3.3.2 Compass Module Testing

To test the compass module, the steps that need to be done are connecting the compass module to the Arduino Mega microcontroller as shown in Figure 3.5 below. In figure 4.6 the compass module is connected with Vcc 3.3V, ground and IC2 pin to sda and scl pins as I2C data communication. At this stage of the test, the author displays the results of the compass sensor readings on the Arduino monitor serial. The mpu9250 module is an optional I2C Master to external 3 sensors. The MPU9250 has a bypass multiplexer interface, which connects the I2C bus system processor (SDA and SCL) directly to the additional I2C sensor bus (AUX_DA and AUX_CL). After the additional sensors are configured by the system processor, the multiplexer bypass interface must be disabled so that the additional I2C master MPU- 9250 can control the I2C bus sensor and collect data from the compass sensor, IMU and barometer sensors. The INT pin must be connected to GPIO on the system processor used to wake the system from suspend mode.

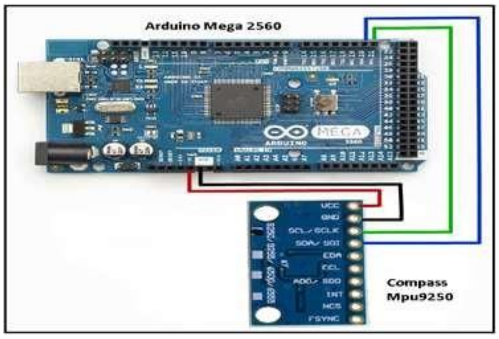


Fig.8 Compass Test Series

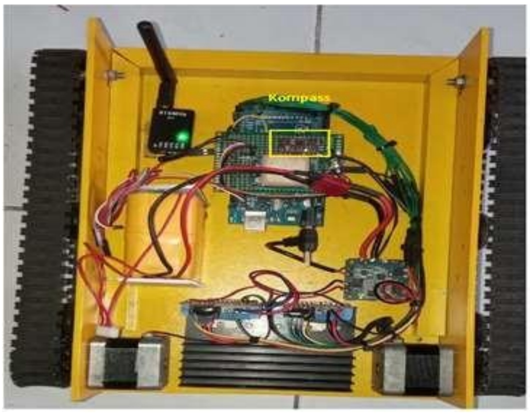


Fig.9 Compass Test Kit

In testing conducted the author connects the device with GCS via telemetry to monitor sensor data through the program interface that has been provided. The test results at 0 ° are presented in the following GCS program interface.

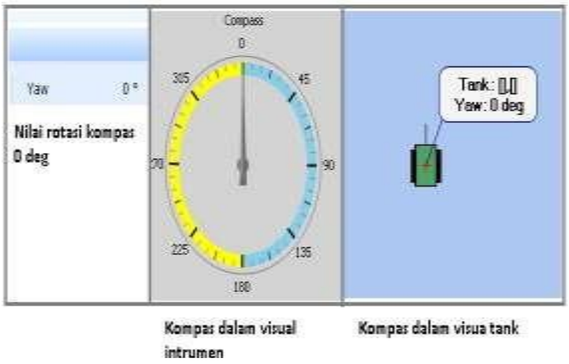


Fig.10 Compass Value Reading Results in GCS

3.3.3 IMU Module Testing

To test the IMU module, the steps that need to be done are connecting the 9250 MPU module with the Arduino Mega microcontroller as shown in Figure 11 In this testing phase, the author displays data from the sensor readings on the pitch and roll axes displayed on the GCS program interface.

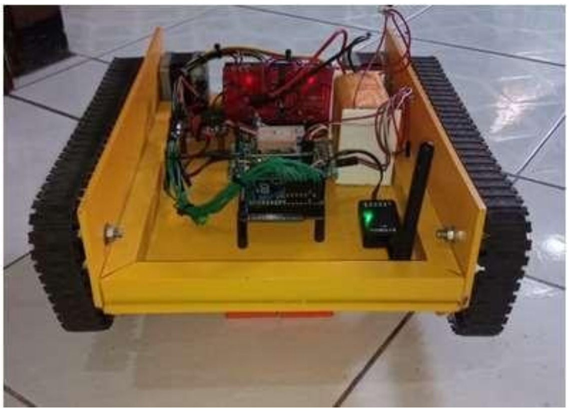


Fig.11 IMU Prototype Tank Test Kit

At the beginning of the IMU sensor test, the author will test the value of the sensor deviation on the roll axis. The device will be tilted from the midpoint to the left and from the point to the right. In the test results the device is placed in the tilt position in Figure 17 below. Then the measurements shown in Figure 12 will be generated.

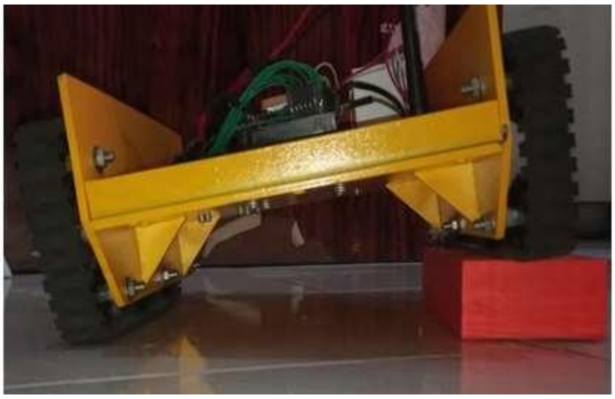


Fig.12 IMU Prototype Tank Test Equipment on Roll Axis

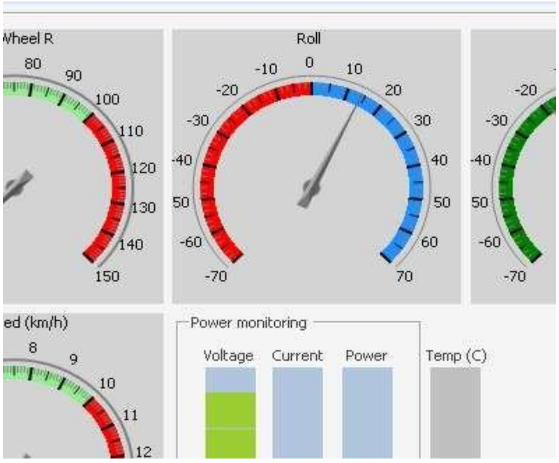


Fig.13 Results of Reading a Roll Value of 10 ° in GCS

The next test the author will change the level of the slope of the device as shown in Figure 14.

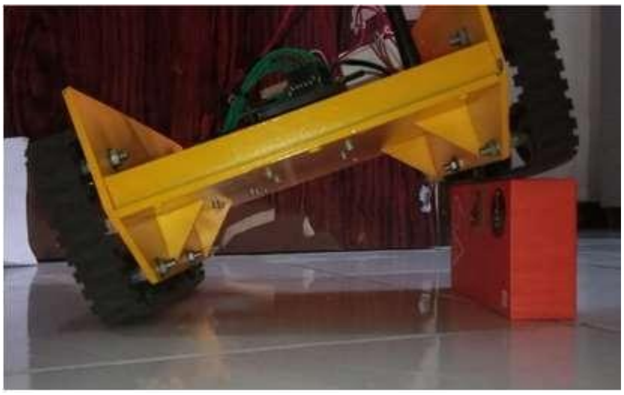


Fig.14 IMU Prototype Tank Test Equipment on Roll Axis

3.3.4 Autopilot Testing System

Autopilot testing, is a system test on the Arduino Mega device running the direction and aim of the prototype tank in carrying out the mission to the target location. In running the autopilot system testing, the system on the Arduino Mega 2560 device needs to be integrated with the Ground Station Control (GCS) system. The commands in running the autopilot control system are obtained from the Ground Station Control (GCS) system. The command is in the form of a latitude and longitude location format as many as the waypoint created. The stages of the autopilot process in the Arduino system can be described as follows:

a. Perform the process of calculating the distance between the location of the prototype tank with the first waypoint.

b. Conduct estimation and identify the distance whether the position of the prototype tank is in the area or minimum radius. If the distance is greater than the minimum radius of the area, the prototype tank will process number three. If the distance of the prototype tank is equal to or smaller than the minimum radius of the area, then the prototype tank is moving towards the waypoint.

c. Perform calculations to find the bow value between the position of the prototype tank and the waypoint.

d. Get yaw values based on compas data.

e. Calculate the angle difference between the bow value and the yaw value, if the yaw value is smaller than the bow value then the prototype tank will move right, if the yaw value is greater than the bow value then the prototype tank will move left. If the yaw angle is the same as the bow value, the prototype tank will move forward.

f. If the prototype tank has reached the waypoint point then the prototype tank will proceed 1,2,3,4,5 and 6 for the next waypoint the entire waypoint has been visited.

The position of the prototype tank can be used as a reference as the location of the ground station control. To be able to recognize the initial location as a ground station control location, an application can be performed to add the ground station location as shown in Figure 15 below.

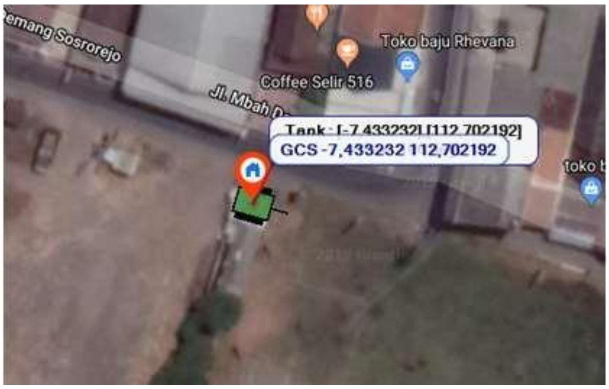


Fig.15 The Location Ground Station Control The

next step, the writer adds the destination waypoint by double clicking on the map of the intended location. The cursor on the map interface will be defined as a location in latitude and longitude format. The following image is the application interface after the authors add the first waypoint.

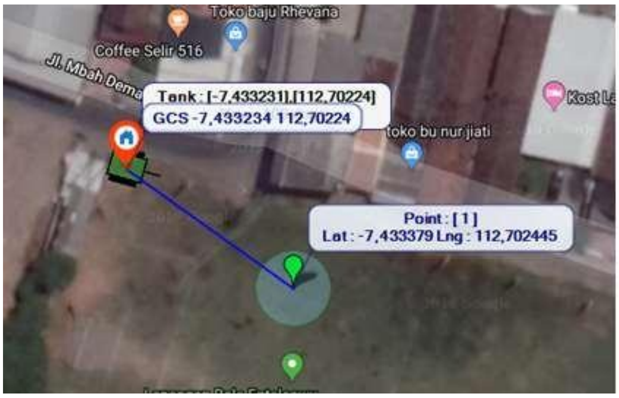


Fig.16 Display Waypoint Interfaces 1

The blue line is a trip route which is a reference point for the prototype tank to waypoint 1. During the journey to waypoint 1 the prototype tank will always update the location data as an error correction if the prototype tank moves not according to the route made. In this test the author will test the auto pilot control system with 1 waypoint. The list of waypoints that have been made can be explained in Table 1 below.

Table 1. Location table and waypoint distance 1

Latitude	Longitude	Distance
-7,43338	112,702445	21,5 m

The distance between GCS and waypoints can be calculated based on the coordinates of the location of the two points. Based on calculations made by the program, the distance to the destination is 21.5 meters.

After the preparation is complete, the writer will upload the waypoint data to the prototype tank and execute the command to start the movement on autopilot.

Next is the interface display of the movement of the prototype tank on the map shown per 5 meters of mileage.

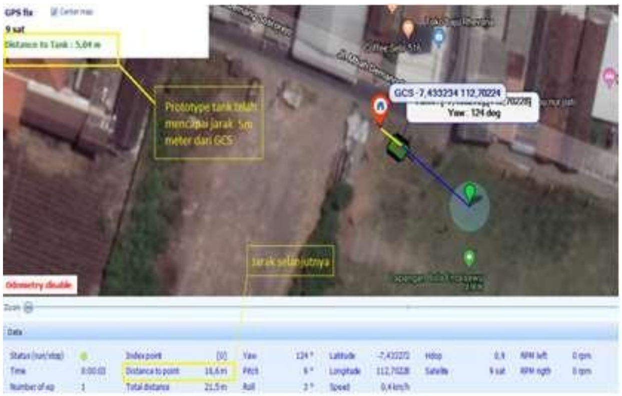


Fig.17 Prototype Tank Has Reached 5m Distance

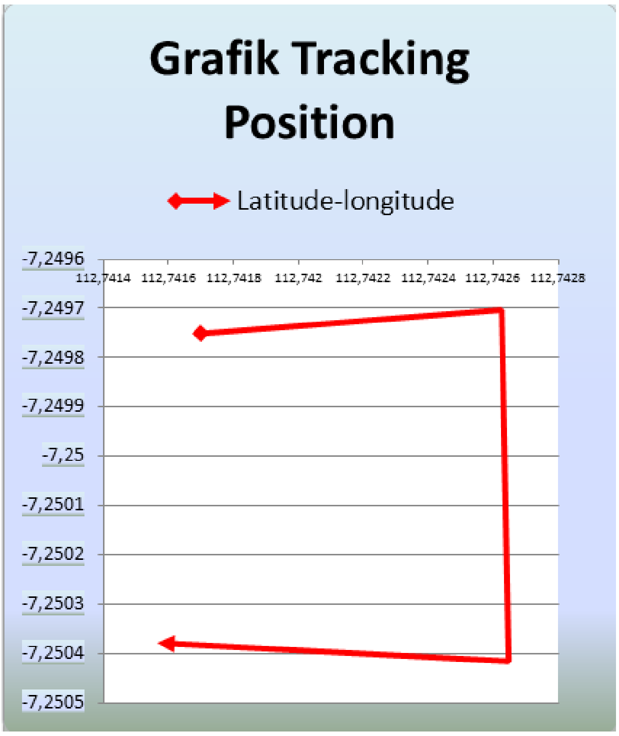


Fig.18 Tank Prototype Position Tracking Chart

4. CONCLUSIONS

The conclusions obtained from The Design of Autopilot Tracking Control at A Prototype Tank Based on Arduino can be stated as follows:

a. Control system that can work on autopilot on a tank vehicle, based on the waypoint points that have been made. The test results show the vehicle tanks can move towards the point of location that has been made. The direction of the tank rides is obtained from the compass bearing value compared to the bow direction calculated based on the difference in the position of the tank rides and the position of the destination location.

b. Based on the manual control test, the response of the tank vehicle in carrying out the manual control with a response of less than 1 second. This is greatly influenced by the control time interval of GCS which always updates data every 1 second.

c. Based on testing that has been done, to access the IMU sensor and compass can be done by accessing it through the I2C communication channel. To access the GPS sensor can be done by accessing via serial communication. GPS data obtained cannot be used immediately, but it is necessary to decode the data to get the original data.

Suggestions that can be given based on the design and research testing of Autopilot Full Build in Arduino Based Prototype Tanks, can be mentioned as follows:

a. Development of location identification uses GPS devices with a higher level of sensitivity than Neo 6M GPS devices, it aims to get better location identification accuracy.

b. The development of the system uses the merging of the GPS sensor with the accelerometer sensor in order to be able to predict locations based on historical location data and accelerate displacement. This method can improve the GPS accuracy factor.

5. REFERENCES

Ahmed Hassan Ahmed, Bahaaeldin Gamal, Ahmed Nasr Ouda, Ahmed Mohsen Kamel and Yehia Zakaria. 2018. Autopilot Design of Unmanned Aerial Vehicles, El-Halwagy Military Technical Research Center, Cairo, Egypt.

Andrianto, Heri and Aan Darmawan. 2016.. Arduino Fast Learning and Programming. Bandung: Bandung Informatics.

ATMEL (2019). 8-bit AVR Microcontroler with 16KbytesIn-SystemProgrammableFlash - ATMega16L. [Pdf] .www.atmel.com / Images / doc2466.pdf.

Adafruit (2014). Adafruit HMC5883L Breakout - Triple-AxisMagnetometer / Compass Sensor. (https://learn.adafruit.com/ada fruit-hmc5883l-breakout-triple-axis-magnetometer-compass-sensor/wiring-and-test, accessed 10 July 2010)

Djuandi, Feri. 2011. "Introduction to Arduino". Jakarta: Elexmedia Arduino Publisher. Arduino Mega 2560.

Kadir, Abdul. 2015. Smart Book Programming Arduino. Yogyakarta MediaKom.

Kadir, Abdul. 2015. A Study Guide for Various Microcontroller-Based Projects. Yogyakarta: ANDI Publisher.

True, Purnomo. 2011. Getting to know I2C Communication (InterIntegratedCircuit). (Http://purnomosejati.wordpress.com/2011/08/25/mengenal-communications-i2cinter-integrated-circuit.

Shantanu Ingle, Madhuri Phute. 2016. Tesla Autopilot: Semi Autonomous Driving, an Uptick for Future Autonomy Department of E&TC, Pune Institute of Computer Technology, Pune, Maharashtra, India.

Theophil Ruzicka. 2017. Model based Design of a Sailboat Autopilot, Embedded and Communication Systems Halmstad University.

.