Conception d’un robot sous-marin type raie manta

- CHAUDAT Jordan
- L2SI
- 2024/2025

- Tuteur : Mr HUGEL
- Laboratoire : COSMER

Plan

Partie	Contenu
Introduction	Contexte / inspiration bio-mimétique
Objectif du projet	Ce que le robot doit accomplire
Conception mécanique	Structure, matériaux, ailes souples
Architecture électronique	Microcontrôleurs, capteurs, alimentation
Programmation et contrôle	Logique embarquée, Wifi, boucles
Plans de tests	Etape de validation, essai en bassin
Résultats attendus	Vitesse, autonomie, stabilité
Application & perspectives	Usages concrets / évolutions futures
Conclusion	Bilan et ouverture

Pourquoi s’inspirer de la raie manta ?

- Nage fluide et silencieuse
- Excellente efficacité énergétique
- Propulsion par ondulation des nageoires
- Grande agilité à faible vitesse
- Adaptée aux environnements complexes

La nature ne gaspille rien, elle optimise tout.

Ce que j'ai voulu reproduire de la raie manta

$$\theta(t) = \theta_0 + A\sin(2\pi f t)$$

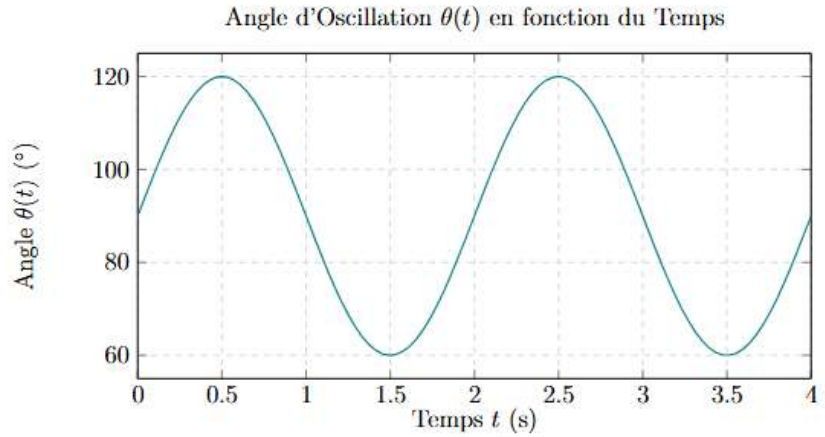
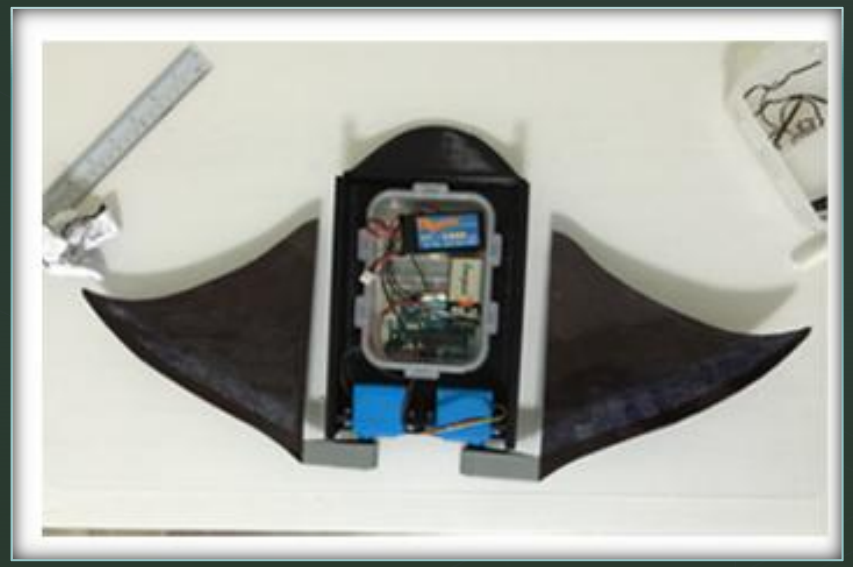


FIGURE 1.1 – Représentation de l'angle d'oscillation des ailes en fonction du temps, avec $\theta_0 = 90°$, $A = 30°$ et $f = 0.5$ Hz.

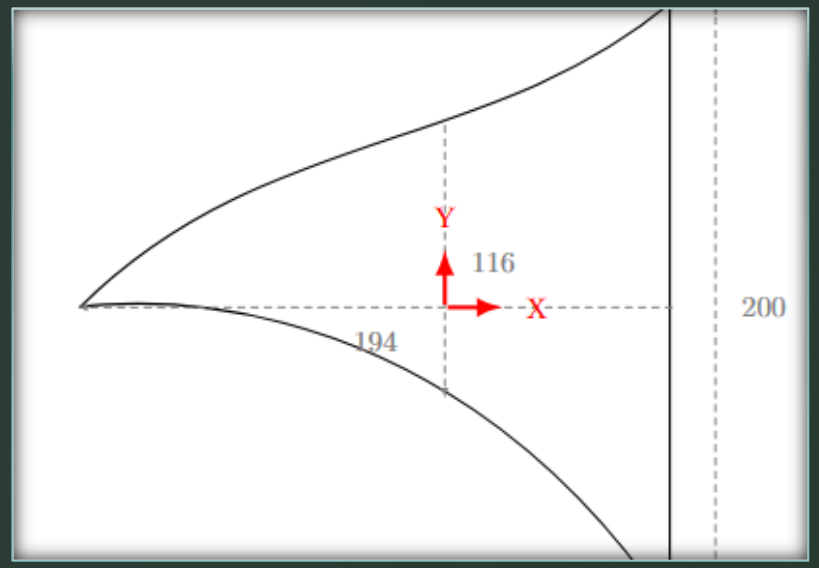
Paramètre	Valeur cible
Amplitude	20 à 40 cm
Fréquence	0.5 à 1.2 Hz
Angle de battement	15 à 30°
Profil du mouvement	Ondulatoire
Souplesse / Déformation	Portance et manoeuvrabilité

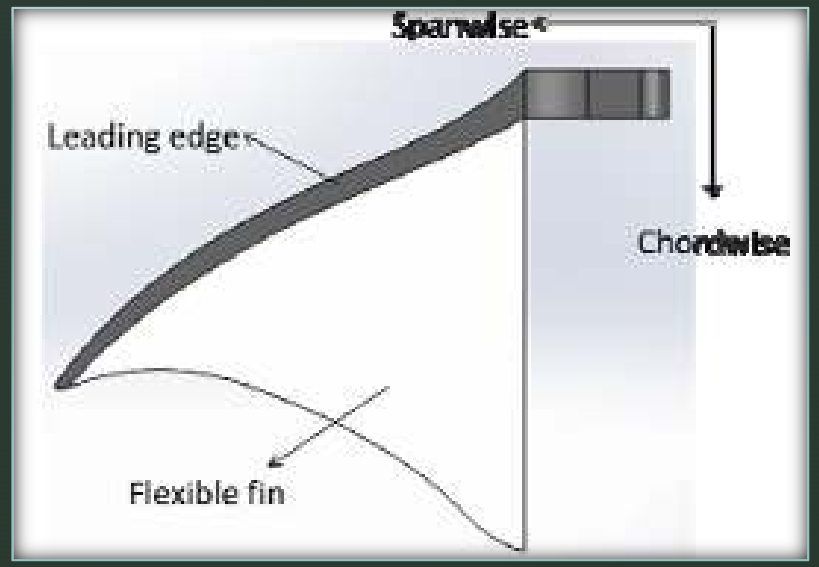
Le corps du robot

- Structure monobloc
- Matériau : nylon PA12
- 630 mm d'envergure, 350 mm de long
- Boitier central étanche
- Modularité



Photographs of Robot Manta Ray prototype, Chee-Meng Chew, Qing-Yuan Lim, K. S. Yeo, National University of Singapore



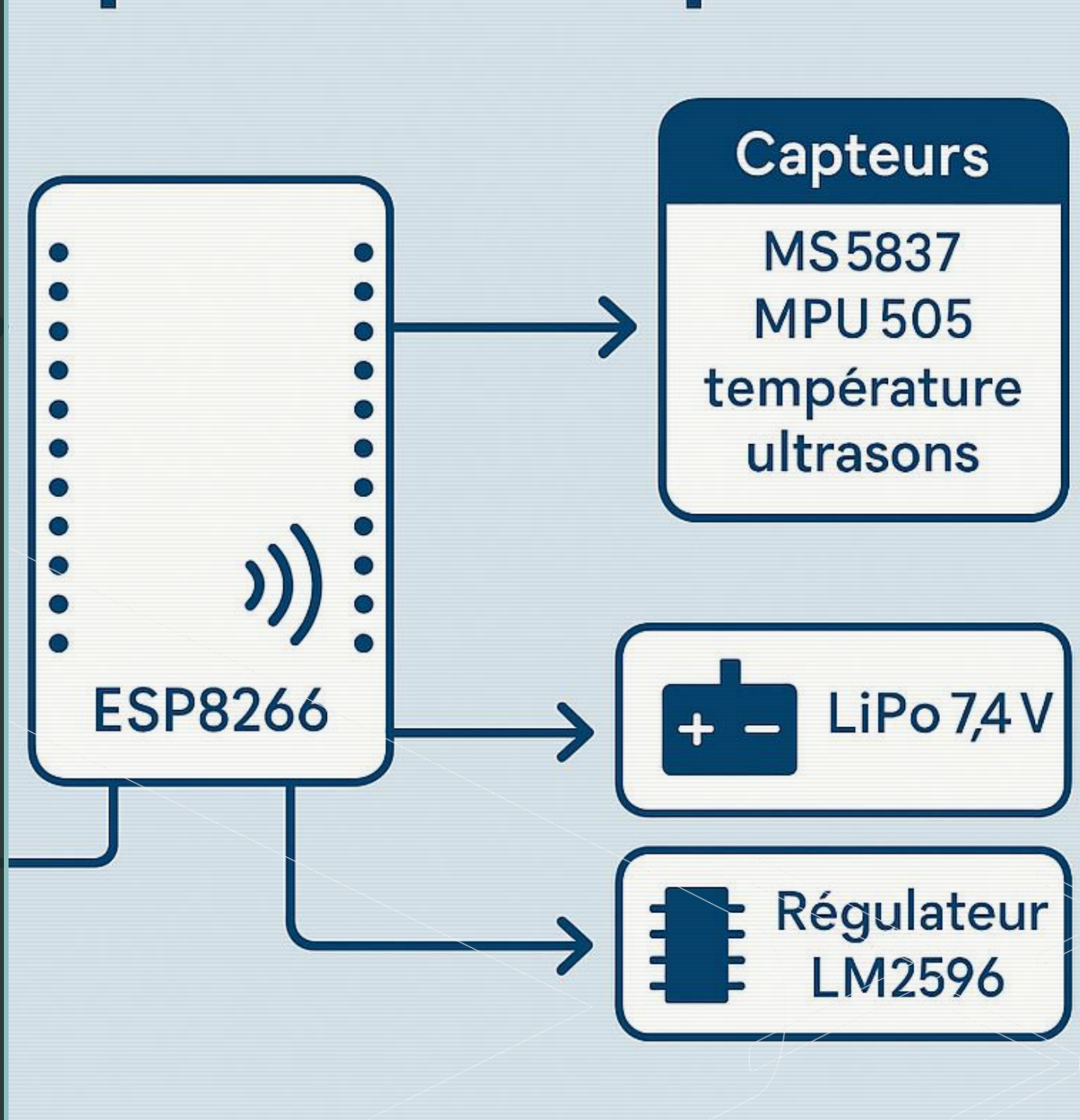


Les ailes flexibles

- Matériau : PVC souple 0,3 mm
- 194 mm x 200 mm
- Fixation : Support oscillant animé par les servos
- Déformation passive

Architecture électronique

- ESP8266 : microcontrôleur Wifi
- MPU6050 : orientation et mouvement
- MS5837 : profondeur
- JSN-SR04T : détection d’obstacles
- DHT22 : température interne
- Batterie LiPo 7.4V – 2200 mAh
- Régulateur LM2596 – 5V stable



Contrôle et programmation

- Servomoteurs pilotés par fonction sinusoidales
- Mouvement opposé des ailes
- Lecture en continu des capteurs
- Interface web embarqué
- Boucles non bloquantes

```
#include <Servo.h>
#include <Wire.h>
#include <WiFi.h>
#include "MPU6050.h"
#include "DHT.h"
#include "MS5837.h"

// 📌 Brochage
#define DHTPIN 4
#define DHTTYPE DHT22
#define echoPin 5
#define trigPin 6
#define pinGauche 9
#define pinDroit 10

// 📡 WiFi
const char* ssid = "RobotManta";
const char* password = "motdepasse";
WiFiServer server(80);

// 🩼 Capteurs
MPU6050 mpu;
DHT dht(DHTPIN, DHTTYPE);
MS5837 pressureSensor;
Servo servoGauche, servoDroit;

// ⏱️ Minuteries
unsigned long lastSensorRead = 0;
unsigned long lastServoUpdate = 0;
const unsigned long sensorInterval = 200;
const unsigned long servoInterval = 20;
```

```
// 🌊 Paramètres de nage
float amplitude = 30; // en degrés
float frequency = 0.5; // Hz
bool active = false;

void setup() {
  Serial.begin(9600);

  // Initialisation capteurs
  Wire.begin();
  mpu.initialize();
  dht.begin();
  pressureSensor.init();
  pressureSensor.setModel(MS5837::MS5837_30BA);
  pressureSensor.setFluidDensity(997); // eau douce

  // Servos
  servoGauche.attach(pinGauche);
  servoDroit.attach(pinDroit);

  // WiFi
  WiFi.softAP(ssid, password);
  server.begin();
}

void loop() {
  unsigned long now = millis();

  // 🔁 Boucle 1 : propulsion
  if (active && now - lastServoUpdate >= servoInterval) {
    lastServoUpdate = now;
    float angle = 90 + amplitude * sin(2 * PI * frequency * now / 1000.0);
    servoGauche.write(angle);
    servoDroit.write(180 - angle);
  }
```

```
// 📈 Boucle 2 : capteurs
if (now - lastSensorRead >= sensorInterval) {
  lastSensorRead = now;
  readSensors();
}

// 🌐 Boucle 3 : WiFi
handleWiFi();


oid readSensors() {
// MPU6050
int16_t ax, ay, az, gx, gy, gz;
mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

// MS5837
pressureSensor.read();
float depth = pressureSensor.depth();

// Ultrason JSN-SR04T
digitalWrite(trigPin, LOW); delayMicroseconds(2);
digitalWrite(trigPin, HIGH); delayMicroseconds(10);
digitalWrite(trigPin, LOW);
long duration = pulseIn(echoPin, HIGH);
float distance = duration * 0.034 / 2;

// DHT22
float temperature = dht.readTemperature();
float humidity = dht.readHumidity();
```

```
  // Affichage console
  Serial.print("Temp: "); Serial.print(temperature);
  Serial.print("°C | Depth: "); Serial.print(depth); Serial.println(" m");
}

void handleWiFi() {
  WiFiClient client = server.available();
  if (client) {
    String request = client.readStringUntil('\r');
    client.flush();

    if (request.indexOf("/start") != -1) active = true;
    if (request.indexOf("/stop") != -1)  active = false;
    if (request.indexOf("/boost") != -1) frequency += 0.1;

    client.println("HTTP/1.1 200 OK");
    client.println("Content-Type: text/html\n");
    client.println("<h1>Robot Manta</h1>");
    client.println("<a href='/start'>Démarrer</a> | <a href='/stop'>Arrêter</a>");
    client.println("<br><a href='/boost'>Boost fréquence</a>");
    client.stop();
  }
}
```

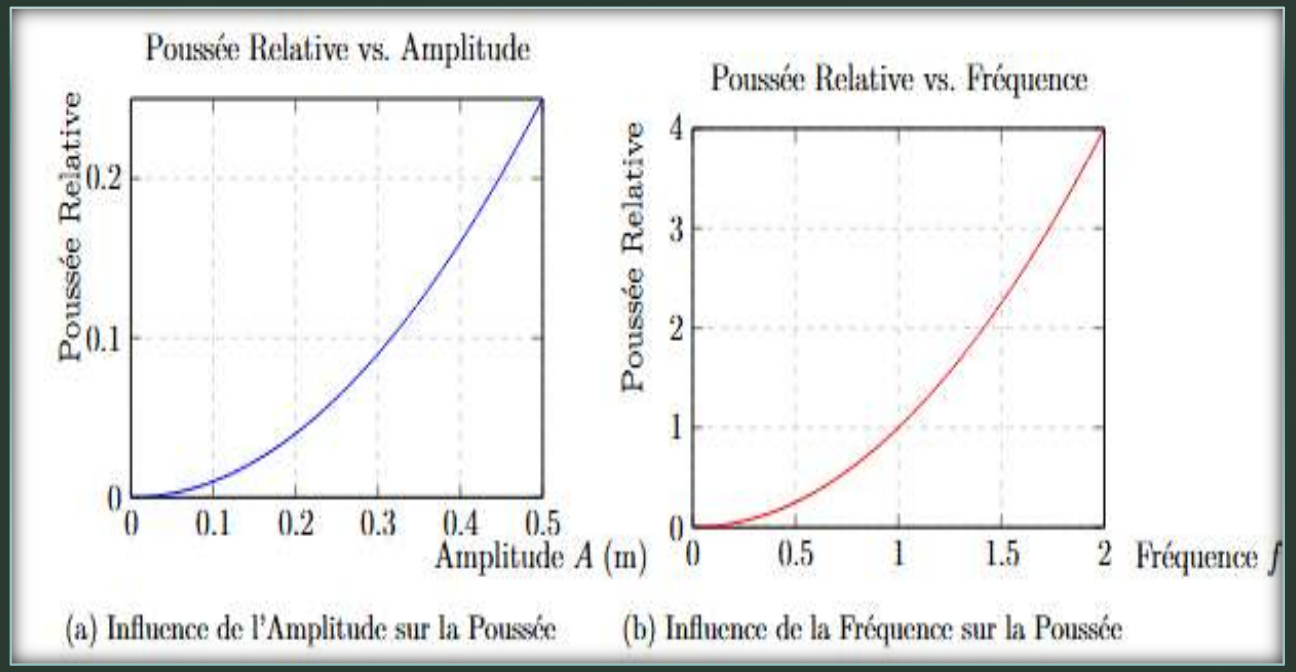
Tests prévus

Servomoteurs	Lecture de capteurs	Etanchéité	Wifi	Essai en bassin
Oscillation fluide	Stabilité des mesures (IMU, pression)	Immersion 30 minutes	Connexion > 5 minutes	Portance et trajectoire
Symétrie des mouvements	Précision des distances (JSN-SR04T)	Aucun dépôt ou fuite	Latence < 500 ms	Mesure de vitesse

Résultat attendu : propulsion

$$F \propto \rho \cdot S \cdot A^2 \cdot f^2$$

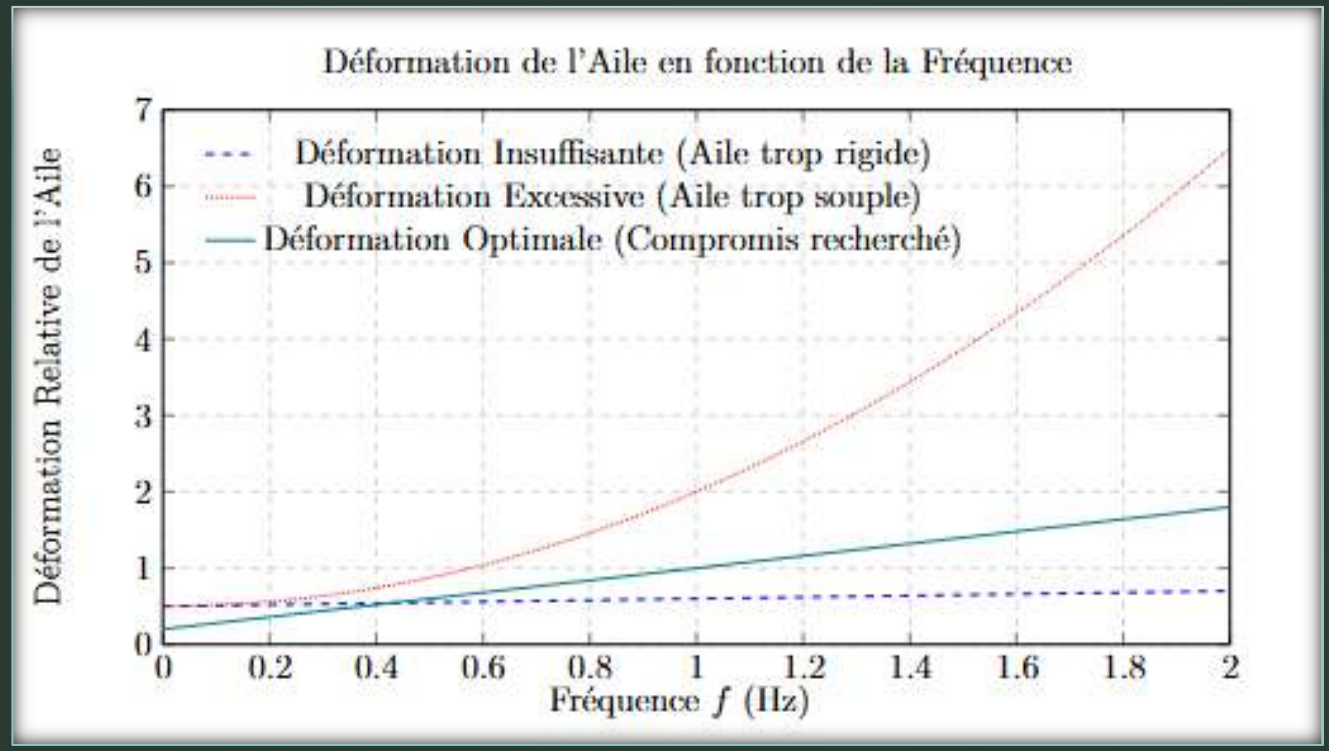
- Poussée estimée : fonction de A^2 et f^2
- Amplitude cible : 20 – 40 cm
- Fréquence optimale : 0.5 – 1.2 Hz
- Vitesse visée : 10 – 15 cm/s



(a) Influence de l'Amplitude sur la Poussée (b) Influence de la Fréquence sur la Poussée

Résultat attendu : déformation des ailes

- Déformation optimale : 15° à 30°
- Souplesse contrôlée
- Compromis parfait entre rigidité - souplesse



Résultat attendu : Autonomie et stabilité

Autonomie = Capacité de la batterie / taux de consommation

Soit ici, 2200 / 1200 = 1,83h = 1h50

- Autonomie estimée : 1h10
- Stabilité fonctionnelle
- Latence < 500 ms
- Trajectoire rectiligne

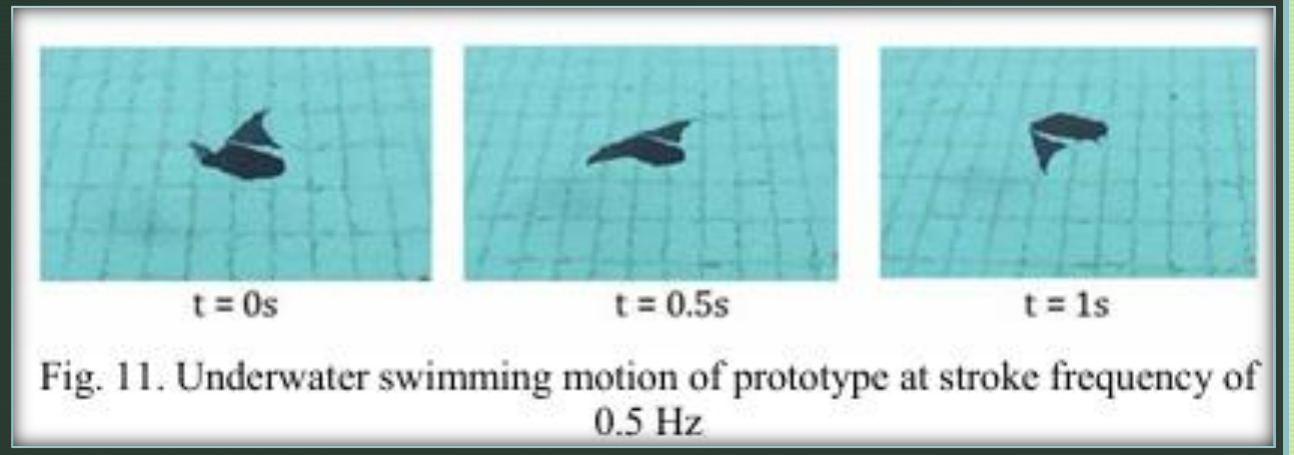


Fig. 11. Underwater swimming motion of prototype at stroke frequency of 0.5 Hz

Photographs of Robot Manta Ray prototype, Chee-Meng Chew, Qing-Yuan Lim, K. S. Yeo, National University of Singapore

Applications & perspectives

Recherche marine

Inspection sous marin

Environnement

Robot Manta : vers quelles applications ?

Education et biomimétisme

Evolution future

Conclusion

- Robot bio-inspiré
- Fonctionnel et modulaire
- Application concrète
- Perspective

Arduino. (2024). *Documentation officielle Servo / WiFi / pulseIn :* https://www.arduino.cc

Topalian, J.-J. (2021). *Vers l'innovation biomimétique : robots adaptables pour les milieux complexes :* https://jeanjacquestopalian.com.

Chee-Meng Chew, Qing-Yuan Lim, K. S. Yeo (2015). Development of Propulsion Mechanism for Robot Manta Ray : Microsoft Word - Robot_Manta_Ray_Conference_paper_Draft_v6.doc

Maheshwari, Vishal, George V. Lauder et Frank E. Fish (2018). Kinematics of swim-ming of the manta ray : three-dimensional analysis of open-water https://jeb.biologists.org/content/221/6/jeb166041

Li, H. et al. (2024). Kinematic and hydrodynamic analysis of a manta ray-inspiredrobot https://www.researchgate.net/publication/391866800_Kinematic_and_hydrodynamic_analysis_of_a_manta_ray-inspired_robot

Hasan, Kayes et al. (2024). Oceanic Challenges to Technological Solutions : A Reviewof Autonomous Underwater Vehicle Path Technologies in Biomimicry, Control, Navi-gation, and Sensing : Oceanic Challenges to Technological Solutions: A Review of Autonomous Underwater Vehicle Path Technologies in Biomimicry, Control, Navigation, and Sensing | IEEE Journals & Magazine | IEEE Xplore

Chee-Meng Chew (2015). NUS-developed manta ray robot swims faster and operates up to 10 hours : NUS-developed manta ray robot swims faster and operates up to 10 hours