Blin Sébastien,
Collin Pierre-Henri,
Louarn Amaury

UNIVERSITÉ DE RENNES 1

Université de Rennes 1

Campus de Beaulieu

Licence STS

Cycle Préparatoire Ingénieur Rennes 1 - Informatique et Télécommunications

---

Rapport de Travail d'Initiative Personnelle Encadrée (TIPE)

Comment la reconnaissance faciale du conducteur peut-elle améliorer sa sécurité au volant ?

Sous l'encadrement de :

Johanne Bézy-Wendling
Maître de Conférences
Responsable cycle préparatoire ingénieur de l'Université de Rennes I
(spécialité informatique et télécommunications)

Finn Jørgensen
Responsable L3 d'informatique - ISTIC

Résumé

Introduction

0.0.1 Pourquoi ce projet

Première partie

Origine du problème

Partie 1

Historique de la reconnaissance faciale

Partie 2

Les enjeux de la sécurité routière

Deuxième partie

Reconnaissance Faciale

Partie 3

Théorie

3.1 Général

3.2 Eigenface

3.3 Fisherface

3.4 LBPH

Partie 4

Expérimentations

4.1 Protocole

4.2 Réalisation

4.3 Résultats et analyse

Troisième partie

Reconnaissance des émotions

Partie 5

Théorie

5.1 Différentes solutions

5.2 Solution choisie

Partie 6

Expérimentations

6.1 Protocole

6.2 Réalisation

6.3 Résultats et analyse

Quatrième partie

Production

Partie 7

Prototype final

Partie 8

Tests finaux

Partie 9

Discussion et analyse

Conclusion

Annexe A

Code des applications

A.1 Application principale

A.2 Code Arduino pour les tests

```
/**
 * oooooooooooo  ooooo  oooooooooo oooooooooooo      o       oooooooooo  oooooooo8
 *  88 888  88    888    888     888 888              888       888    888 888
 *     888        888    888oooo88  888ooo8        8  88      888oooo88  888oooooo
 *     888        888    888         888           8oooo88     888               888
 *    o888o      o888o  o888o       o888ooo8888 o88o  o888o o888o       o88ooo888
 */

int pinEngine = 7;
int pinWarning = 6;
int pinAccelerationLimit = 5;
int pinBrake = 4;
int pinSound = 8;

boolean engine = false;

char command = 0;

void Exit()
{
  engine = false;
  digitalWrite(pinEngine,LOW);
  digitalWrite(pinWarning,LOW);
  digitalWrite(pinAccelerationLimit,LOW);
  digitalWrite(pinBrake,LOW);

  noTone(pinSound);
}

void Engine()
{
  engine = true;
  digitalWrite(pinEngine, HIGH);
}

void Warning()
{
```

```
38    digitalWrite(pinWarning, HIGH);
39  }
40
41  void StopWarning()
42  {
43    digitalWrite(pinWarning, LOW);
44  }
45
46  void AccelerationLimit()
47  {
48    digitalWrite(pinAccelerationLimit, HIGH);
49  }
50
51  void StopAccelerationLimit()
52  {
53    digitalWrite(pinAccelerationLimit, LOW);
54  }
55
56  void Brake()
57  {
58    digitalWrite(pinBrake, HIGH);
59  }
60
61  void StopBrake()
62  {
63    digitalWrite(pinBrake, LOW);
64  }
65
66  void Sound()
67  {
68    tone(pinSound, 666);
69  }
70
71  void StopSound()
72  {
73    noTone(pinSound);
74  }
75
76  void setup()
77  {
78    pinMode(pinEngine, OUTPUT);
79    pinMode(pinWarning,OUTPUT);
80    pinMode(pinAccelerationLimit,OUTPUT);
81    pinMode(pinBrake,OUTPUT);
82
83    Serial.begin(9600); //On demarre la connexion serie
84    while(!Serial){} // on attend que la connexion serie demarre
85
86    /**
87     *  verification du fonctionnement des systemes
88     *  (1/2 seconde)
89     */
90    digitalWrite(pinEngine, HIGH);
91    Warning();
92    AccelerationLimit();
93    Brake();
94    Sound();
95
96    delay(500);
97
98    digitalWrite(pinEngine, LOW);
99    StopWarning();
```

```
100     StopAccelerationLimit();
101     StopBrake();
102     StopSound();
103   }
104
105   void loop()
106   {
107     if(!engine)//si non demarre
108     {
109       command = Serial.read();
110       if(command == 'e')
111         Engine();
112       command = 0;
113     }
114     else
115     {
116       if(Serial.available() > 0) //si on recoit une donnee sur le port serie
117       {
118         command = Serial.read();
119         switch(command)
120         {
121           case 'x':
122             Exit();
123           break;
124           case 'w':
125             Warning();
126           break;
127           case 'r':
128             StopWarning();
129           break;
130           case 'b':
131             Brake();
132           break;
133           case 'n':
134             StopBrake();
135           break;
136           case 'a':
137             AccelerationLimit();
138           break;
139           case 'q':
140             StopAccelerationLimit();
141           break;
142           case 's':
143             Sound();
144           break;
145           case 'd':
146             StopSound();
147           break;
148         }
149         command = 0;
150       }
151     }
152   }
```