

```

1  /*****
2  * Descrição: Arquivo header das funções responsáveis
3  * pelo controle de voo do drone.
4  * Autores: Gustavo L. Fernandes e Giácomo A. Dollevedo
5  * Última Atualização: 6/01/2021
6  *****/
7  #ifndef FlightControl_h
8  #define FlightControl_h
9
10 #include "Arduino.h"
11 #include "IMU.h"
12 #include <analogWrite.h>
13 #include <ESP32PWM.h>
14 #include <ESP32Servo.h>
15 #include <ESP32Tone.h>
16
17 #define PIDMAX 400
18
19 typedef struct kPID
20 {
21 float fkp, fki, fkd;
22 };
23
24 class FlightControl
25 {
26
27     private:
28         kPID _gains;
29         float _fsetPoint;
30         float _fpidCalculated;
31         float _fimem;
32         float _ferrorPrevious;
33         char _axis;
34         float _setPoint;
35
36
37     public:
38     /* *****/
39     /* Nome do metodo:          FlighControl  - Construtor Vazio
40     /*
41     /* Descrição:              Inicializa um objeto controlador padrão, ou seja
42     /*                          com valores de ganho nulos
43     /*                          Setpoint nulo e constante
44     /*
45     /* Parametros de entrada:  Nenhum (Vazio)
46     /*
47     /* Parametros de saida:    Nenhum (Vazio)
48     /*

```

```

48  /*
    */
49  /*
    */
50  /* *****
    */
51      FlightControl(char Axis);
52
53  /* *****
    */
54  /* Nome do metodo:          FlighControl  - Construtor Pre Definido
    */
55  /* Descrição:              Inicializa um objeto controlador com ganhos definidos
    */
56  /*
    */
57  /*
    */
58  /* Parametros de entrada: (float fkp, float fki, float fkd) ganhos dos
    controladores*/
59  /*                          proporcional, integrador e derivativo
    */
60  /*                          Setpoint nulo e constante
    */
61  /*                          (char Axis) podendo ser "r", "p", "y" para determinar
    */
62  /*                          qual o eixo que o objeto será definido
    */
63  /*
    */
64  /* Parametros de saida: Nenhum (Vazio)
    */
65  /*
    */
66  /*
    */
67  /* *****
    */
68      FlightControl(float fkp, float fki, float fkd, char Axis);
69
70  /* *****
    */
71  /* Nome do metodo:          pidVelControl
    */
72  /* Descrição:              - Rotina de Controle de velocidade
    */
73  /*                          para um determinado eixo
    */
74  /*
    */
75  /*
    */
76  /* Parametros de entrada: (float axisVel) variavel que contem os valores medidos
    */
77  /* de velocidade lidos da IMU que será dado como entrada pro controlador
    */
78  /* Parametros de saida: Nenhum (Vazio)
    */

```

```

79  /*
80  /*
81  /* *****
82      void pidVelControl(float  axisVel);
83
84  /* *****
85  /* Nome do metodo:          getPID_Calculated
86  /* Descrição:              - retorna o valor de PID calculado a ser utilizado
87  /*                          para saida do sistema
88  /*
89  /* Parametros de entrada: Nenhum (Vazio)
90  /*
91  /* Parametros de saida: float _pidCalculated
92  /*
93  /*
94  /* *****
95      float getPID_Calculated();
96
97  /* *****
98  /* Nome do metodo:          getGains
99  /* Descrição:              - retorna o valor dos ganhos do controlador PID
100 /*
101 /*
102 /* Parametros de entrada: Nenhum (Vazio)
103 /*
104 /* Parametros de saida: kPID _gains
105 /*
106 /*
107 /* *****
108      kPID getGains();
109
110 /* *****

```

```

111 /* Nome do metodo:          setKp
112 */
113 /* Descrição:              - Seta um novo valor para o ganho do controlador
114 */
115 /*                          proporcional
116 */
117 /* Parametros de entrada: float kp
118 */
119 /*
120 /* *****
121 */
122 void setKp(float kp);
123
124 /* *****
125 */
126 /* Nome do metodo:          setKd
127 */
128 /* Descrição:              - Seta um novo valor para o ganho do controlador
129 */
130 /*                          derivativo
131 */
132 /* Parametros de entrada: float kd
133 */
134 /*
135 /* Parametros de saida: Nenhum (Vazio)
136 */
137 /*
138 /* *****
139 */
140 void setKd(float kd);
141
142 /* *****
143 */
144 /* Nome do metodo:          setKi
145 */
146 /* Descrição:              - Seta um novo valor para o ganho do controlador
147 */
148 /*                          integrativo
149 */
150 /*
151 /* Parametros de entrada: float ki
152 */
153 /*

```

```

143  /*
    */
144  /* Parametros de saida: Nenhum (Vazio)
    */
145  /*
    */
146  /*
    */
147  /* *****
    */
148      void setKi(float ki);
149
150  /* *****
    */
151  /* Nome do metodo:          setSetPoint
    */
152  /* Descrição:              - Seta um novo valor para o setPoint do controlador
    */
153  /*                          de velocidade
    */
154  /*
    */
155  /* Parametros de entrada: float newSetPoint
    */
156  /*
    */
157  /* Parametros de saida: Nenhum (Vazio)
    */
158  /*
    */
159  /*
    */
160  /* *****
    */
161      void setSetPoint(float newSetPoint);
162
163  /* *****
    */
164  /* Nome do metodo:          getSetPoint
    */
165  /* Descrição:              - Consulta o atual valor para o setPoint do
    */
166  /*                          controlador*/
    */
167  /*
    */
168  /* Parametros de entrada: Nenhum (Vazio)
    */
169  /*
    */
170  /* Parametros de saida: float _setPoint
    */
171  /*
    */
172  /*
    */
173  /* *****
    */
174      float getSetPoint();

```

```

175 /* *****
176 */
177 /* Nome do metodo:          getAccError
178 */
179 /* Descrição:              - Consulta o erro acumulado do controlador
180 */
181 /* Parametros de entrada: Nenhum (Vazio)
182 */
183 /* Parametros de saida: float _fimem
184 */
185 */
186 /* *****
187 */
188     float getAccError();
189
190 /* *****
191 */
192 /* Nome do metodo:          getPreviousError
193 */
194 /* Descrição:              - Consulta o erro previo  do controlador
195 */
196 /* Parametros de entrada: Nenhum (Vazio)
197 */
198 /* Parametros de saida: float _ferrorPrevious
199 */
200 /* *****
201 */
202     float getPreviousError();
203
204 };
205
206
207
208
209
210 #endif
211

```

