

```

1  /*****
2  * Descrição: Arquivo c++ que implementa o controlador de voo do drone.
3  * Autores: Gustavo L. Fernandes e Giácomo A. Dollevedo
4  * Ultima Atualização: 6/01/2021
5  *****/
6  #include "FlightControl.h"
7  #include "ThrottleControl.h"
8  #include "Arduino.h"
9  #include "IMU.h"
10 #include <analogWrite.h>
11 #include <ESP32PWM.h>
12 #include <ESP32Servo.h>
13 #include <ESP32Tone.h>
14
15 /* *****/
16 /* Nome do metodo:          FlighControl  - Construtor Vazio
17 /*
18 /* Descrição:              Inicializa um objeto controlador padrão, ou seja
19 /*
20 /*                          com valores de ganho nulos
21 /*
22 /*                          Setpoint nulo e constante
23 /*
24 /* Parametros de entrada: Nenhum (Vazio)
25 /*
26 /*
27 /* Parametros de saída: Nenhum (Vazio)
28 /*
29 /*
30 /* *****/
31
32 FlightControl::FlightControl(char Axis)
33 {
34
35 //Ganho de cada um dos controladores, proporcional, integrativo e derivativo
36     _gains.fkp = 0;
37     _gains.fki = 0;
38     _gains.fkd = 0;
39     _axis = Axis;
40     _fimem = 0;
41
42 //Set point de entrada para o controlador (Inicial = 0)
43     _setPoint = 0;
44 }
45
46 /* *****/
47 */

```

```

45  /* Nome do metodo:          FlighControl - Construtor Pre Definido
   */
46  /* Descrição:              Inicializa um objeto controlador com ganhos definidos
   */
47  /*
   */
48  /*
   */
49  /* Parametros de entrada: (float fkp, float fki, float fkd) ganhos dos
   controladores*/
50  /*          proporcional, integrador e derivativo
   */
51  /*          Setpoint nulo e constante
   */
52  /*          (char Axis) podendo ser "r", "p", "y" para determinar
   */
53  /*          qual o eixo que o objeto será definido
   */
54  /*
   */
55  /* Parametros de saida: Nenhum (Vazio)
   */
56  /*
   */
57  /*
   */
58  /* *****
   */
59
60  FlightControl::FlightControl(float fkp, float fki, float fkd, char Axis)
61  {
62
63  //Ganho de cada um dos controladores, proporcional, integrativo e derivativo
64      _gains.fkp = fkp;
65      _gains.fki = fki;
66      _gains.fkd = fkd;
67      _axis = Axis;
68      _fimem = 0;
69      _ferrorPrevious = 0;
70
71  //Set point de entrada para o controlador (Inicial = 0)
72      _setPoint = 0;
73  }
74
75
76  /* *****
   */
77  /* Nome do metodo:          pidVelControl
   */
78  /* Descrição:              - Rotina de Controle de velocidade
   */
79  /*          para um determinado eixo
   */
80  /*
   */
81  /*
   */
82  /* Parametros de entrada: (float axisVel) variavel que contem os valores medidos
   */

```

```

83  /* de velocidade lidos da IMU que será dado como entrada pro controlador
    */
84  /* Parametros de saida: Nenhum (Vazio)
    */
85  /*
    */
86  /*
    */
87
88
89 void FlightControl::pidVelControl(float axisVel)
90 {
91     //YAW = Z
92     //ROLL = Y
93     //PITCH = X
94     //Taxas de variação angular em cada um dos eixos da IMU
95     float fgyro;
96
97     //Testa qual eixo o objeto foi relacionado para coletar a informação adequada da IMU
98     fgyro = axisVel;
99
100
101     //Erro instantaneo e acumulado
102     float ferror_temp = fgyro - _setPoint;
103     _fimem += _gains.fki * ferror_temp;
104
105     //limitando o sinal de erro acumulado
106
107     if(_fimem > PIDMAX) {
108         _fimem = PIDMAX;
109     }
110     else if(_fimem < (PIDMAX * -1)){
111         _fimem = PIDMAX * -1;
112     }
113
114     //Sinal de saida do PID e atualização do erro previo
115
116     _fpidCalculated = _gains.fkp*ferror_temp + _fimem + _gains.fkd*(ferror_temp -
    _ferrorPrevious);
117
118     //limitando o limite de ajuste possível
119
120     if(_fpidCalculated > PIDMAX) {
121         _fpidCalculated = PIDMAX;
122     }
123     else if(_fpidCalculated < (PIDMAX * -1)){
124         _fpidCalculated = PIDMAX * -1;
125     }
126
127     _ferrorPrevious = ferror_temp;
128 }
129
130 /* *****
    */
131 /* Nome do metodo:          getPID_Calculated
    */
132 /* Descrição:              - retorna o valor de PID calculado a ser utilizado
    */

```

```

133  /*                                     para saída do sistema
134  */
135  /* Parametros de entrada: Nenhum (Vazio)
136  */
137  /* Parametros de saída: float _pidCalculated
138  */
139  /*
140  /* *****
141  */
142
143  float FlightControl::getPID_Calculated(){
144      return _fpidCalculated;
145  }
146
147  /* *****
148  /* Nome do metodo:          getGains
149  /* Descrição:              - retorna o valor dos ganhos do controlador PID
150  /*
151  /*
152  /* Parametros de entrada: Nenhum (Vazio)
153  /*
154  /* Parametros de saída: kPID _gains
155  /*
156  /*
157  /* *****
158  */
159  kPID FlightControl::getGains(){
160      return _gains;
161  }
162
163  /* *****
164  /* Nome do metodo:          setKp
165  /* Descrição:              - Seta um novo valor para o ganho do controlador
166  /*                          proporcional
167  /*

```

```
168  /* Parametros de entrada: float kp
    */
169  /*
    */
170  /* Parametros de saida: Nenhum (Vazio)
    */
171  /*
    */
172  /*
    */
173  /* *****
    */
174
175  void FlightControl::setKp(float kp){
176      _gains.fkp = kp;
177  }
178
179  /* *****
    */
180  /* Nome do metodo:          setKd
    */
181  /* Descrição:              - Seta um novo valor para o ganho do controlador
    */
182  /*                          derivativo
    */
183  /*
    */
184  /* Parametros de entrada: float kd
    */
185  /*
    */
186  /* Parametros de saida: Nenhum (Vazio)
    */
187  /*
    */
188  /*
    */
189  /* *****
    */
190  void FlightControl::setKd(float kd){
191      _gains.fkd = kd;
192  }
193  }
194
195  /* *****
    */
196  /* Nome do metodo:          setKi
    */
197  /* Descrição:              - Seta um novo valor para o ganho do controlador
    */
198  /*                          integrativo
    */
199  /*
    */
200  /* Parametros de entrada: float ki
    */
201  /*
    */
```

```

202  /* Parametros de saida: Nenhum (Vazio)
    */
203  /*
    */
204  /*
    */
205  /* *****
    */
206
207 void FlightControl::setKi(float ki){
208     _gains.fki = ki;
209
210 }
211
212
213  /* *****
    */
214  /* Nome do metodo:          setSetPoint
    */
215  /* Descrição:              - Seta um novo valor para o setPoint do controlador
    */
216  /*                          de velocidade
    */
217  /*
    */
218  /* Parametros de entrada: float newSetPoint
    */
219  /*
    */
220  /* Parametros de saida: Nenhum (Vazio)
    */
221  /*
    */
222  /*
    */
223  /* *****
    */
224
225 void FlightControl::setSetPoint(float newSetPoint){
226     _setPoint = newSetPoint;
227 }
228
229
230  /* *****
    */
231  /* Nome do metodo:          getSetPoint
    */
232  /* Descrição:              - Consulta o atual valor para o setPoint do
    controlador*/
233  /*
    */
234  /*
    */
235  /* Parametros de entrada: Nenhum (Vazio)
    */
236  /*
    */
237  /* Parametros de saida: float _setPoint
    */

```

```

238 /*
239 */
240 /* *****
241 */
242 float FlightControl::getSetPoint(){
243     return _setPoint;
244 }
245
246 /* *****
247 */
248 /* Nome do metodo:          getAccError
249 */
250 /* Descrição:              - Consulta o erro acumulado do controlador
251 */
252 /*
253 */
254 /* Parametros de entrada: Nenhum (Vazio)
255 */
256 /* Parametros de saida: float _fimem
257 */
258 /*
259 */
260 float FlightControl::getAccError(){
261     return _fimem;
262 }
263
264 /* *****
265 */
266 /* Nome do metodo:          getPreviousError
267 */
268 /* Descrição:              - Consulta o erro previo  do controlador
269 */
270 /*
271 */
272 /* Parametros de entrada: Nenhum (Vazio)
273 */
274 /* Parametros de saida: float _ferrorPrevious
275 */
276 /*
277 */
278 /*
279 */
280 /*
281 */
282 /*
283 */
284 /*
285 */
286 /*
287 */
288 /*
289 */
290 /*
291 */
292 /*
293 */
294 /*
295 */
296 /*
297 */
298 /*
299 */
300 /*
301 */
302 /*
303 */
304 /*
305 */
306 /*
307 */
308 /*
309 */
310 /*
311 */
312 /*
313 */
314 /*
315 */
316 /*
317 */
318 /*
319 */
320 /*
321 */
322 /*
323 */
324 /*
325 */
326 /*
327 */
328 /*
329 */
330 /*
331 */
332 /*
333 */
334 /*
335 */
336 /*
337 */
338 /*
339 */
340 /*
341 */
342 /*
343 */
344 /*
345 */
346 /*
347 */
348 /*
349 */
350 /*
351 */
352 /*
353 */
354 /*
355 */
356 /*
357 */
358 /*
359 */
360 /*
361 */
362 /*
363 */
364 /*
365 */
366 /*
367 */
368 /*
369 */
370 /*
371 */
372 /*
373 */
374 /*
375 */
376 /*
377 */
378 /*
379 */
380 /*
381 */
382 /*
383 */
384 /*
385 */
386 /*
387 */
388 /*
389 */
390 /*
391 */
392 /*
393 */
394 /*
395 */
396 /*
397 */
398 /*
399 */
400 /*
401 */
402 /*
403 */
404 /*
405 */
406 /*
407 */
408 /*
409 */
410 /*
411 */
412 /*
413 */
414 /*
415 */
416 /*
417 */
418 /*
419 */
420 /*
421 */
422 /*
423 */
424 /*
425 */
426 /*
427 */
428 /*
429 */
430 /*
431 */
432 /*
433 */
434 /*
435 */
436 /*
437 */
438 /*
439 */
440 /*
441 */
442 /*
443 */
444 /*
445 */
446 /*
447 */
448 /*
449 */
450 /*
451 */
452 /*
453 */
454 /*
455 */
456 /*
457 */
458 /*
459 */
460 /*
461 */
462 /*
463 */
464 /*
465 */
466 /*
467 */
468 /*
469 */
470 /*
471 */
472 /*
473 */
474 /*
475 */
476 /*
477 */
478 /*
479 */
480 /*
481 */
482 /*
483 */
484 /*
485 */
486 /*
487 */
488 /*
489 */
490 /*
491 */
492 /*
493 */
494 /*
495 */
496 /*
497 */
498 /*
499 */
500 /*
501 */
502 /*
503 */
504 /*
505 */
506 /*
507 */
508 /*
509 */
510 /*
511 */
512 /*
513 */
514 /*
515 */
516 /*
517 */
518 /*
519 */
520 /*
521 */
522 /*
523 */
524 /*
525 */
526 /*
527 */
528 /*
529 */
530 /*
531 */
532 /*
533 */
534 /*
535 */
536 /*
537 */
538 /*
539 */
540 /*
541 */
542 /*
543 */
544 /*
545 */
546 /*
547 */
548 /*
549 */
550 /*
551 */
552 /*
553 */
554 /*
555 */
556 /*
557 */
558 /*
559 */
560 /*
561 */
562 /*
563 */
564 /*
565 */
566 /*
567 */
568 /*
569 */
570 /*
571 */
572 /*
573 */
574 /*
575 */
576 /*
577 */
578 /*
579 */
580 /*
581 */
582 /*
583 */
584 /*
585 */
586 /*
587 */
588 /*
589 */
590 /*
591 */
592 /*
593 */
594 /*
595 */
596 /*
597 */
598 /*
599 */
600 /*
601 */
602 /*
603 */
604 /*
605 */
606 /*
607 */
608 /*
609 */
610 /*
611 */
612 /*
613 */
614 /*
615 */
616 /*
617 */
618 /*
619 */
620 /*
621 */
622 /*
623 */
624 /*
625 */
626 /*
627 */
628 /*
629 */
630 /*
631 */
632 /*
633 */
634 /*
635 */
636 /*
637 */
638 /*
639 */
640 /*
641 */
642 /*
643 */
644 /*
645 */
646 /*
647 */
648 /*
649 */
650 /*
651 */
652 /*
653 */
654 /*
655 */
656 /*
657 */
658 /*
659 */
660 /*
661 */
662 /*
663 */
664 /*
665 */
666 /*
667 */
668 /*
669 */
670 /*
671 */
672 /*
673 */
674 /*
675 */
676 /*
677 */
678 /*
679 */
680 /*
681 */
682 /*
683 */
684 /*
685 */
686 /*
687 */
688 /*
689 */
690 /*
691 */
692 /*
693 */
694 /*
695 */
696 /*
697 */
698 /*
699 */
700 /*
701 */
702 /*
703 */
704 /*
705 */
706 /*
707 */
708 /*
709 */
710 /*
711 */
712 /*
713 */
714 /*
715 */
716 /*
717 */
718 /*
719 */
720 /*
721 */
722 /*
723 */
724 /*
725 */
726 /*
727 */
728 /*
729 */
730 /*
731 */
732 /*
733 */
734 /*
735 */
736 /*
737 */
738 /*
739 */
740 /*
741 */
742 /*
743 */
744 /*
745 */
746 /*
747 */
748 /*
749 */
750 /*
751 */
752 /*
753 */
754 /*
755 */
756 /*
757 */
758 /*
759 */
760 /*
761 */
762 /*
763 */
764 /*
765 */
766 /*
767 */
768 /*
769 */
770 /*
771 */
772 /*
773 */
774 /*
775 */
776 /*
777 */
778 /*
779 */
780 /*
781 */
782 /*
783 */
784 /*
785 */
786 /*
787 */
788 /*
789 */
790 /*
791 */
792 /*
793 */
794 /*
795 */
796 /*
797 */
798 /*
799 */
800 /*
801 */
802 /*
803 */
804 /*
805 */
806 /*
807 */
808 /*
809 */
810 /*
811 */
812 /*
813 */
814 /*
815 */
816 /*
817 */
818 /*
819 */
820 /*
821 */
822 /*
823 */
824 /*
825 */
826 /*
827 */
828 /*
829 */
830 /*
831 */
832 /*
833 */
834 /*
835 */
836 /*
837 */
838 /*
839 */
840 /*
841 */
842 /*
843 */
844 /*
845 */
846 /*
847 */
848 /*
849 */
850 /*
851 */
852 /*
853 */
854 /*
855 */
856 /*
857 */
858 /*
859 */
860 /*
861 */
862 /*
863 */
864 /*
865 */
866 /*
867 */
868 /*
869 */
870 /*
871 */
872 /*
873 */
874 /*
875 */
876 /*
877 */
878 /*
879 */
880 /*
881 */
882 /*
883 */
884 /*
885 */
886 /*
887 */
888 /*
889 */
890 /*
891 */
892 /*
893 */
894 /*
895 */
896 /*
897 */
898 /*
899 */
900 /*
901 */
902 /*
903 */
904 /*
905 */
906 /*
907 */
908 /*
909 */
910 /*
911 */
912 /*
913 */
914 /*
915 */
916 /*
917 */
918 /*
919 */
920 /*
921 */
922 /*
923 */
924 /*
925 */
926 /*
927 */
928 /*
929 */
930 /*
931 */
932 /*
933 */
934 /*
935 */
936 /*
937 */
938 /*
939 */
940 /*
941 */
942 /*
943 */
944 /*
945 */
946 /*
947 */
948 /*
949 */
950 /*
951 */
952 /*
953 */
954 /*
955 */
956 /*
957 */
958 /*
959 */
960 /*
961 */
962 /*
963 */
964 /*
965 */
966 /*
967 */
968 /*
969 */
970 /*
971 */
972 /*
973 */
974 /*
975 */
976 /*
977 */
978 /*
979 */
980 /*
981 */
982 /*
983 */
984 /*
985 */
986 /*
987 */
988 /*
989 */
990 /*
991 */
992 /*
993 */
994 /*
995 */

```

```
273 /* *****  
274 */  
275 float FlightControl::getPreviousError(){  
276     return _ferrorPrevious;  
277 }  
278
```