Úvod

Tato ročníková práce se zabývá sestavením drona tak. Jedním z mých cílů bylo přistupovat k úkolu tak, abych co nejvíce částí vytvořil sám bez použití prodaných polotovarů. Toto splňuje například rám drona, který jsem navrhnul a následně vytiskl na 3D tiskárně.

Projekt je postaven na mikrokontroleru Arduino, který jsem zvolil pro jeho univerzální schopnosti integrace se senzory a ovládání modulů.

Moje motivace pro tuto ročníkovou práci bylo vyzkoušet si řešení nové sady problému, se kterou jsem se dříve nesetkal a kterým se v běžné školní výuce nevěnujeme.

Dron

Dron je obecně definován jako bezpilotní letoun. Tato práce se zabývá konkrétně dronem pro civilní účely. Tyto drony se běžně používají k rekreačním účelům. Převážně pak k natáčení videí z výšky.

Komponenty dronu

Gyroskop a akcelerometr:

Gyroskop a akcelerometr jsou součástky potřebné k vyvažování drona. Jejich funkce je měření úhlu náklonu drona vůči vektoru tíhového zrychlení. K tomuto účelu jsem využil modul MPU6050, který v sobě obsahuje jak gyroskop, tak i akcelerometr. Propojení s arduinem je snadné, jelikož modul využívá pro komunikaci sériovou sběrnici I2C.

Gyroskop měří rotační úhel v závislosti na čase. když k této hodnotě měříme také uplynulý čas, můžeme získat celý úhel natočení za danou časovou jednotku. sčítáním této hodnoty v cyklu pak lze získat celkový úhel natočení vůči původní poloze. Takto by to mohlo fungovat pouze teoreticky v ideálním případě, kdyby nedocházelo k žádné chybě při měření a výpočtu. V reálném případě však k nepřesnostem dochází a výsledná chyba se nasčítává, což za krátkou dobu vede k nezanedbatelné chybě měření. Proto je třeba tuto chybu kompenzovat další metodou měření. V tomto případě akcelerometrem.

Akcelerometr měří směrový vektor zrychlení. Když žádným směrem nezrychluje, lze použít na měření vektoru tíhového zrychlení. V ideálním případě šel tento modul použít k měření úhlu náklonu samostatně. Ovšem hodnoty, které vrací, nedosahují dostatečné přesnosti a responsivity. Dále jsou pak skreslené v případě, že dron zrychluje směrem, který není rovnoběžný s vektorem tíhového zrychlení

Řešením, které se pro tuto aplikaci obecně používá je kombinace těchto dvou složek, při které akcelerometr v určitém poměru kompenzuje chybu vzniklou u gyroskopu. Lze tak docílit relativně přesného a responzivního měření použitelého k vyvažování drona.

software pro gyroskop a akcelerometr:

K ovládání modulu MPU6050 je na internetu dostupných mnoho knihoven pro Arduino. Mnoho z nich jsem vyzkoušel, ovšem žádná z nich mi nevyhovovala zcela. Hlavní problém jsem měl s rychlostí čtení dat, s formátem čtení dat a s prvotní kalibrací. Proto jsem k tomuto účelu napsal vlastní knihovnu za pomoci dokumentace modulu a několika dostupných příkladů kódu.

V první fázi je třeba modul zkalibrovat a to jak pro gyroskop, tak pro akcelerometr. Kalibrace se vykoná zavoláním funkce initiate .Kalibrace gyroskopu probíhá tak, že se v cyklu měří průměrná hodnota vrácená gyroskopem. Jedná se o relativně konstantní chybu, kterou je třeba při dalším jednoduše odečíst od měřené hodnoty. Tuto kalibraci lze spolehlivě provádět pouze ve stavu bez pohybu drona (bez rotace po osách X, nebo Y)

U akcelerometru probíhá kalibrace podobně, ovšem naměřená hodnota se pak prohlásí za současný úhel natočení drona.

Dále funkce refresh aktualizuje hodnoty náklonu. Funkci refresh je vhodné volat co nejčastěji pro minimalizaci chyby vzniklé v gyroskopu. funkce počítá, jak úhel náklon, tak i změnu náklonu v závislosti na čase. úhel náklonu je potřeba pro algoritmus vyvažování PID. Do třídy gyroskopu a akcelerometru (mpu) jsem naimplementoval možnost čtení aktuální hodnoty změny náklonu v závislosti na čase, jelikož je tato hodnota potřeba při výpočtu derivační složky vyvažovacího algoritmu PID. Běžně se tato hodnota počítá vydělením změny časem. Tím, že moje knihovny umožňují pracovat přímo s hodnotou vracenou senzorem bez vzniku dalších výpočetních chyb jsem docílil větší přesnosti a rychlosti celého systému. Toto byla jedna z mých motivací pro naprogramování mých vlastních knihoven pro modul MPU6050 a počítání s algoritmem PID.

PID regulátor:
PID regulátor je algoritmus obecně využívaný k vyvažování. K fungování potřebuje nějakou formu vstupu a výstupu. Příkladem může být termostat, kde se od naměřené teploty odvíjí intenzita zahřívání. Tento algoritmu se také využívá k vyvažování dronů, kde se síla jednotlivých motorku odvíjí od úhlu náklonu.
Tento algoritmus počítá tři složky, které se v závěru sečtou ve výslednou hodnotu:

- proporcionální
- integrální
- derivační

Váha těchto jednotlivých složek se pak určuje třemi konstantami, které je u každého systému potřeba vyladit.

Proporcionální složka se počítá jednoduše tak, že se adekvátní konstantou vynásobí aktuální hodnota náklonu. Čím více je dron nakloněn, tím větší je tato složka.

Derivační složka vychází z okamžité změny v náklonu. Běžně se počítá jako rozdíl posledních dvou hodnot náklonu vydělen změnou v čase. Já jsem však použil přímo hodnotu rozdílu v náklonu vracenou gyroskopem. Tato hodnota je také vynásobena zvolenou konstantou. Funkce této složky je především vybalancování prudších změn v náklonu.

Integrální složka se počítá sečtením rozdílů náklonu od vyrovnané polohy opět vynásobených konstantou. Tato složka adresuje případ, kdy na soustavu působí nějaká stálá síla. Například když dron není dokonale vyvážen, nebo když fouká vítr

Bluetooth komunikace:
Pro ovládání drona jsem zvolil využití bluetooth z prostého důvodu, a to takového, že jsem měl již k dispozici modul pro bluetooth komunikaci. Tudíž to pro mě v danou situaci představovalo nejjednodušší řešení. Osobně však nevidím velký rozdíl mezi jinými alternativami, jako je například rádiová komunikace, nebo wifi, z hlediska integrace s arduinem. Bluetooth modul má také tu výhodu, že lze ovládat za pomoci mobilního telefonu.

Pro komunikaci jsem tedy využil bluetooth modul HC-05, který umožňuje jednoduché propojení s arduinem za pomocí sériové komunikace. V průběhu jsem však narazil na několik potíží ohledně spolehlivosti, které bylo třeba vyřešit.

Při užití jakékoliv bezdrátové komunikace je třeba počítat s možností výpadku spojení a ztráty informace. Bylo tedy třeba implementovat způsob kontroly příchozích informací pro případ, že došlo k vypadnutí, nebo neúplného přijetí příkazů. Komunikace mezi Arduinem a telefonem probíhá tak, že telefon odesílá příkazy v podobě 3 bytů. 1. byte představuje intenzitu motorů a zbývající 2 znamenají náklon v osách X a Y. příkaz telefon odešle při každé změně a také pravidelně za určitý časový interval. V případě, že arduino nedostane žádný příkaz během časového intervalu delšího, tak může říci, že bylo spojení nějakým způsobem přerušeno. Dále v případě, že dostane méně než 3 byty a zbytek bytů z příkazu nepřijde do určitého časového limitu, tak neúplný příkaz vypustí.

Motorky
K ovládání brushless DC slouží moduly ESC (Electronic speed control). za pomoci tichto modulů je arduino schopno ovládat rychlost, kterou se motorky otáčejí. Moduly, které jsem použil vyžadují k řízení PWM signál o frekvenci 50 Hz, který je arduino schopno vygenerovat za pomoci knihovny Servo.h. Před každým startem je třeba kalibrovat ESC moduly tím, že se nejprve na 2 sekundy nastaví PWM signál na maximální hodnotu a následně po dobu dalších 2 sekund na hodnotu minimální. Zvolený typ motoru je A2212/6T 2200KV. Z mé zkušenosti se mohou motorky stejného typu od jiného dodavatele částečně lišit.

Plastové díly
Použitý rám jsem navrhnul v programu **Fusion 360**. Následně jsem vytvořil G-code v programu **Slic3r** a vytiskl na 3D tiskárně **Anet A8 PLUS** s použitím materiálu PLA. Hlavní část rámu, na který jsou upevněny motorky, tvoří jeden díl tištěný v celku. Dále jsou součástí ochranná svodidla připevněná pod motorky a dvě plastová patra oddělená vytištěnými oddělovači. První patro náleží elektronice a kabeláží. V druhém patře upevněna baterie. Během testovacích letů jsem se 2krát setkal s komplikacemi, při kterých se mi hlavní část rámu zlomila v nejslabším místě u motorků. Problém jsem vyřešil zesílením a prodloužením výztuží, které jsou umístěny podél obvodu rámu. Díky tomu jsem také snížil ohebnost a mohl jsem si tak dovolit snížit procento infilu, což mělo za následek snížení celkové hmotnosti.

Popis konstrukce

PID tester

Komplikace

Můj počátění přístup byl takový, že jsem se snažil vyladit PID hodnoty testováním přímo v terénu, což mělo za následek mnoho pádů a několik poškozených dílů. Nakonec se mi sice podařilo dosáhnout krátkého letu s takto získanými hodnotami pro PID, ale po následném pádu jsem se raději rozhodl pro další ladění sestavit testovací zařízení.
Během práce na projektu jsem se k němu snažil přistupovat s respektem, jelikož jsem si byl vědom možných nebezpečí. Při každém pokusu jsem například nosil ochranné brýle a více vrstev oblečené. Při jednom nezdařeném pokusu, kdy nastala chyba v komunikaci (proti které jsem pak následně vytvořil několik různých opatření), jsem byl lehce posekán vrtulkou do prstů.

Závěr

https://www.electronoobs.com/eng_robotica_tut9_2.php
https://howtomechatronics.com/tutorials/arduino/arduino-brushless-motor-control-tutorial-esc-bldc/
https://www.arduino.cc/en/Tutorial/Foundations/PWM
https://howtomechatronics.com/tutorials/arduino/arduino-and-hc-05-bluetooth-module-tutorial/
https://appinventor.mit.edu/
https://en.wikipedia.org/wiki/PID_controller
https://en.wikipedia.org/wiki/Electronic_speed_control
https://www.autodesk.com/products/fusion-360/overview
https://slic3r.org/
https://www.thingiverse.com/thing:993593