Autopilot*

Samuel Švec
Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
xsvecs@stuba.sk

14.12.2021

Abstrakt

Tento článok sa bude venovať histórií a softvérovom fungovaní autopilota v lietadlách a taktiež v automobilovom priemysle. Rozoberie aj aktuálne chyby v softvéroch, ktoré môžu ovplyvniť spoľahlivosť a bezpečnosť autopilota a tým aj zastaviť používanie daného softvéru.

Kľúčové slová: autopilot, história, softvér, Tesla, aplikácie

1 Úvod

Autopilot nie je v dnešnej dobe nový pojem. S autopilotom sa vieme stretnúť aj v našom bežnom živote, a to napríklad v leteckej či cestnej doprave. V oblasti automobilového priemyslu však nie je dostatočne autopilot využívaný. V súčasnej dobe niekoľko popredných firiem pracuje na vytvorení dokonalého automobilového autopilota, ako napríklad Tesla. Prvý autopilot existuje už vyše sto rokov. Náznaky autonómneho letu pri lietadlách a autonómnej jazdy sú však ešte ďaleko. Jeden z problémov, ktorý bol naznačený v úvode, je podrobnejšie vysvetlený v častiach2.1 a 3.1. Ďalej je to rozvinuté v častiach4 a 5

2 Autopilot v leteckej doprave

2.1 História autopilota v letectve

Autopilot je technický systém, ktorý je počas prevádzky schopný zastúpiť človeka v riadení ovládaného objektu bez ďalšej ľudskej asistencie. Historicky prvý funkčný autopilot bol zostrojený v roku 1912 Lawrencom Sperrym, predstavený však bol verejnosti prvý raz až v roku 1914 a uvedený do prevádzky v lodnej doprave. [3] Na obr. 1 môžeme vidieť vynálezcu Lawrenca Sperryho.

*Semestrálny projekt v predmete Metódy inžinierskej práce, ak. rok 2021/22, vedenie: Vladimír Mlynarovič

Obr. 1: Lawrence Sperry (naľavo) - prvý funkčný autopilot [5]

2.2 Autopilot v lietadlách

V letectve sa často stretávame s pomenovaním AFCS.[1] Tieto systémy sú univerzálne používané v komerčnom letectve. Rozsiahle uplatnenie nachádzajú aj vo vojenskom a všeobecnom letectve, ale koncepcia voľného letu, v rámci ktorej bude v budúcnosti prebiehať plne automatický let, je v podstate zameraná na zníženie preťaženia dýchacích ciest, čo je situácia, ktorá ovplyvňuje najmä komerčné letectvo. Hoci lietadlá všeobecného letectva wm musia tiež pracovať v prostredí voľného letu, väčšina týchto lietadiel má buď iba manuálne riadiace systémy, alebo je inštalácia AFCS len základná. [1]

Hlavným účelom použitia AFCS je do určitej miery automatizovať lietanie lietadla, aby sa znížila pracovná záťaž pilotov (zvyčajne v určitej kritickej fáze letu), aby sa zachovala bezpečnosť letu. Čoraz častejšie sa AFCS používajú aj na zlepšenie základných letových vlastností lietadla (napr. na zabezpečenie dynamickej stability, aj keď bolo lietadlo navrhnuté ako staticky nestabilné), alebo na overenie základných výkonov lietadla v niektorých atmosférických podmienkach. Na dosiahnutie plne automatického letu bude potrebné, aby sa dosiahlo niekoľko dôležitých technologických a operačných systémových vývojov, ale vždy, keď sa to podarí, výsledný plne automatický systém bude fungovať prostredníctvom už vyvinutého AFCS. [2]

3 Autopilot v automobiloch

3.1 Počiatky autopilota v automobiloch značky Tesla

text bude pridaný vo finálnej verzii

3.2 Tesla Autopilot v dnešných dňoch

text bude pridaný vo finálnej verzii

[1] Automatic Flight Control Systems - automatické systémy riadenia letu

Porovnanie SW verzií Autopilota	Autopilot 1 (2014-2016)	Full Self Driving (2021+)
Adaptívny tempomat	Áno	Áno
Automatické riadenie	Áno	Áno
Funkcia privolania auta	Áno	Áno
Inteligentné privolanie	Nie	Áno
Automatické parkovanie	Áno	Áno
Automatická zmena jazdného pruhu	Áno	Áno
Navigácia pomocou autopilota	Nie	Áno
Rozoznanie značky Stop a semafórov	Nie	Áno
Automatické riadenie v uliciach mesta	Nie	Áno

Obr. 2: Tabuľka - Porovnanie softvérových verzií autopilota [6]

4 Adaptívne riadenie distribuovaných aplikácií autopilota

Aj keď sa programovacie modely aj paralelné počítačové systémy naďalej rýchlo vyvíjajú, väčšina analýz výkonnosti zostáva založená na procese vyvinutom pred viac ako štyridsiatimi rokmi:

1. *Aplikačné prístrojové vybavenie* - Aplikačný kód môže byť vybavený nástrojmi automaticky alebo manuálne vložením volaní do rutín knižnice nástrojov. Počas následného vykonávania knižnica nástrojov zaznamenáva príslušné údaje o výkone, vrátane počtu a časov vykonávania procedúr, slučiek a základných blokov.

2. *Extrakcia údajov o výkone* - Po inštrumentácii sa zachytia údaje o výkone z jedného alebo viacerých vykonávaní programu. V ideálnom prípade tieto spustenia zahŕňajú vstupné údaje a výpočtové zdroje typické pre tie, ktoré sa vyskytujú v produkčnom prostredí.

3. *Analýza a vizualizácia* - Po následnom spracovaní sa údaje o výkone vizualizujú a analyzujú, aby sa identifikovali úzke miesta výkonu aplikačného programu (napr. pomocou textových profilovacích nástrojov alebo vizualizačných systémov ako AIMS alebo Pablo)

4. *Optimalizácia aplikácie* - Na základe merania a analýzy sa buď program upraví tak, aby sa zmiernili vnímané úzke miesta, alebo sa prispôsobia pravidlá runtime systému tak, aby lepšie zodpovedali požiadavkám na zdroje programu. [4]

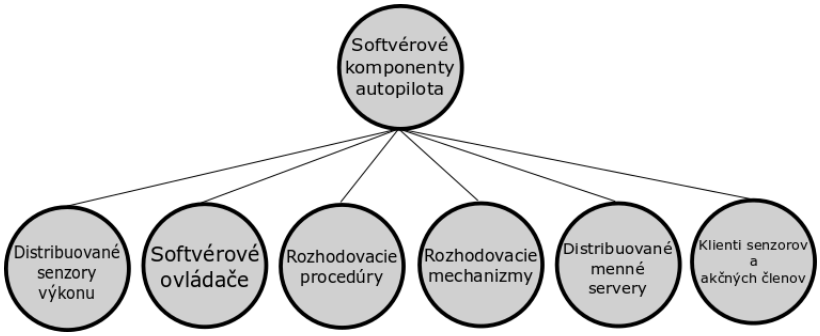
4.1 Softvérové komponenty autopilota

Oddelením merania výkonu, riadenia a rozhodovania umožňuje Autopilot systémovým dizajnérom nahradiť softvérové rozhodovacie postupy vizualizáciou v reálnom čase a interaktívnym riadením, keď rýchlosť zmien pripúšťa ľudskú kontrolu.

Akýkoľvek adaptívny riadiaci systém musí monitorovať príslušné stavy systému, určiť, aké zmeny sú potrebné, a tieto zmeny realizovať, aby sa splnili požadované ciele. Aby sa dynamicky optimalizovalo správanie aplikácií a behu

systému pre distribuované výpočtové siete, systém adaptívneho výkonu s uzavretou slučkou musí zahŕňať niečo z nasledujúceho:

- *Distribuované senzory výkonu* - dokážu zachytiť údaje o výkone aplikácií a systému a generovať popisy požiadaviek na zdroje a metriky výkonu
- *Softvérové ovládače* - môžu povoliť a konfigurovať správanie aplikácií a zásady správy zdrojov
- *Rozhodovacie procedúry* - ako lokálne, tak aj globálne, na výber politík správy zdrojov a aktivovanie akčných členov na základe pozorovaných požiadaviek na zdroje aplikácií a systémových odoziev zachytených senzormi výkonu.
- *Rozhodovacie mechanizmy* - využívajú údaje z distribuovaných senzorov na vyváženie často protichodných optimalizačných cieľov
- *Distribuované menné servery* - podporujú registráciu vzdialených senzorov a akčných členov a požiadavky na senzory a akčné členy založené na vlastnostiach vzdialených klientov
- *Klienti senzorov a akčných členov* - interagujú so vzdialenými senzormi a akčnými členmi, monitorujú dáta senzorov a vydávajú príkazy akčným členom [4]



Obr. 3: Diagram - Softvérové komponenty autopilota [4]

5 Chyby autopilota

Bezpilotné lietadlá (UAV) sú v modernej spoločnosti čoraz dôležitejšie a široko používané. Softvérové chyby v týchto systémy môžu spôsobiť vážne problémy, ako sú zlyhania systému, zamrznutie a nedefinované správanie. Existuje mnoho štúdií o chybách v rôznych typoch softvéru, avšak vlastnosti softvérových chýb UAV neboli nikdy systematicky skúmané. To bráni vývoju nástrojov na zabezpečenie spoľahlivosti UAV. Vykonaná bola rozsiahla empirická štúdia na dvoch známych softvérových platformách autopilota s otvoreným zdrojom pre UAV, a to konkrétne PX4 a Ardupilot, v ktorej cieľom bolo charakterizovať chyby v UAV. Prostredníctvom analýzy bolo nájdených osem typov chýb špecifických pre UAV. V následujúcej podkapitole budú opísané. [7]

5.1 Príčiny chýb autopilota v bezpilotných lietadlách

Podľa štúdie sa príčiny chýb autopilota delia na:

- *Limit* - Systém UAV je často kompatibilný s množstvom rôzneho hardvéru, a preto je sprevádzaný množstvom hardvérových obmedzení. Napríklad PX4 má 1 306 limitov. V praxi je pre vývojárov ťažké správne zvládnuť veľké množstvo limitov. Keď urobia chyby, UAV systémy môžu trpieť rôznymi typmi limitných chýb.

- *Matematika* - Systémy UAV sa často spoliehajú na rôznych komplexných riadiacich a odhadovacích algoritmoch. V porovnaní s tradičným softvérom je softvér UAV náchylnejší na matematické chyby. Niekedy môže byť pre vývojárov veľmi ťažké presne vybrať najvhodnejší matematický vzorec.

- *Nedôslednosť* - Chyby často vznikajú, keď vývojári nie sú oboznámení s konzistentnosťou medzi hardvérom a softvérom v systéme UAV. Typickým prípadom je, že vývojári nesprávne používajú funkciu s nesprávnymi modelmi dronov. Okrem nezrovnalostí medzi funkciami a modelmi dronov existujú aj nezrovnalosti medzi hardvérovými rozhraniami a protokolmi rozhraní, nezrovnalosti medzi senzormi a knižnicami atď. Chyby nesúladu sa častejšie vyskytujú v systéme UAV navrhnutom pre viacero zariadení a môžu spôsobiť zlyhanie dronu, keď kritické funkcie zlyhajú.

- *Priorita* - Na rozdiel od tradičných softvérových prioritných chýb sú niektoré prioritné chyby špecifické pre UAV spôsobené prioritou hardvéru. Tieto typy chýb je ťažké odpozorovať, pretože v logike programu nie sú žiadne zjavné chyby a väčšina takýchto chýb nespôsobuje výraznú odchýlku výkonu systému. Náplasť, ktorá opravuje chybu, mení poradie premeny tlaku a teploty.

- *Parameter* - Parametre v systéme UAV sú veľmi komplikované. Napríklad PX4 má viac ako 1000 parametrov. Väčšina parametrov obsahuje limit a predvolenú hodnotu, ktorú definujú vývojári na základe príslušného atribútu funkčného modulu. Keďže nastavenia parametrov ovplyvňujú výkon systému UAV, nesprávne zaobchádzanie s parametrami môže spôsobiť chyby.

- *Hardvérová podpora* - V tradičných softvérových systémoch sa vyskytujú chyby hardvérovej podpory. Chyby hardvérovej podpory v systémoch UAV sa nelíšia od chýb v tradičných systémoch. Väčšina z nich je spôsobená poruchami vodiča. Tento typ chýb sa nazýva UAVspecific, pretože podiel takýchto chýb v systéme UAV je veľký, keďže hardvérová podpora UAV nie je vo všeobecnosti taká dobrá ako v tradičnom systéme.

- *Oprava* - Údaje získané niektorými snímačmi je potrebné pred ich použitím opraviť. Napríklad údaje GPS môžu byť narušené rôznymi podmienkami prostredia (napr. teplotou), a preto je potrebné ich opraviť, aby sa zabezpečila presnosť. Keďže v systéme UAV sú potrebné intenzívne opravy údajov, vývojári môžu ľahko zmeškať nejaký opravný proces, čo má za následok rôzne neočakávané chyby.

- *Inicializácia* - Podobne ako pri oprave údajov bolo zistené, že chýbajúca inicializácia je tiež typickou chybou vývojárov. Dokonca aj niektorí vývojári pamätajú na vykonanie inicializácie, ale často môžu zabudnúť vykonať inicializáciu (t. j. resetovať hodnoty) počas výpočtu. [7]

6 Zhrnutie

Literatúra

[1] Said D. Jenie and Agus Budiyono. Automatic flight control system. Bandung, Indonesia, January 2006.

[2] Donald McLean. Automatic flight control systems. *Measurement and Control*, 36:172–175, July 2003.

[3] Robert C. Nelson. Flight stability and automatic control. 1989.

[4] R.L. Ribler, J.S. Vetter, H. Simitci, and D.A. Reed. Autopilot: adaptive control of distributed applications. In *Proceedings. The Seventh International Symposium on High Performance Distributed Computing (Cat. No.98TB100244)*, pages 172–179, 1998.

[5] William Scheck. Lawrence sperry: Genius on autopilot. `https://www.historynet.com/lawrence-sperry-autopilot-inventor-and-aviation-innovator.htm`.

[6] Radovan Skokan. Všetko o funkčnosti systému tesla autopilot na slovensku: Reálne situácie, porovnanie verzií. `https://www.mojelektromobil.sk/system-autonomnej-jazdy-tesla-aktopilot-na-slovensku-historia-vyvoj/`.

[7] Dinghua Wang, Shuqing Li, Guanping Xiao, Yepang Liu, and Yulei Sui. An exploratory study of autopilot software bugs in unmanned aerial vehicles. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2021, page 20–31, New York, NY, USA, 2021. Association for Computing Machinery.