FLIGHT SIMULATOR BASED ON JASON MULTI-AGENT SYSTEM

Pavlína Punčochářová
Bachelor Degree Programme (4), FIT BUT
E-mail: xpunco01@stud.fit.vutbr.cz

Supervised by: Jan Samek
E-mail: samejan@fit.vutbr.cz

Abstract: This thesis is focused on Jason system and AgentSpeak programming language. The main goal is to create the flight simulator driven by plan based on the Jason system. Planes are represented by agents and each agent is able to resolve conflicts, such as collision with another agent, which can occur during the simulation.

Keywords: Jason, AgentSpeak, agent, flight simulator

1 ÚVOD

Cílem této práce je prostudování multi-agentního systému Jason a jazyka AgentSpeak a možnosti jejich využití pro vytvoření 2D leteckého simulátoru. Jednotlivá letadla jsou reprezentována agenty a jejich chování je popsáno plány v jazyce AgentSpeak. Agenti jsou schopni reagovat na aktuální stav prostředí aplikováním vhodných plánů s cílem doletět do koncového bodu, případně řešit vzniklé konflikty (například vyhnutí se kolizi, pronásledování nepřátelského letadla, apod.).

2 NÁVRH A IMPLEMENTACE SIMULÁTORU

Protože se jedná o simulaci, tedy o prostředek simulující realitu, je i vizuální podoba práce navržena tak, aby se blížila realitě. Simulátor je vykreslen v podobě radaru, na kterém jsou jednotlivá letadla zobrazena jako pohybující se body spolu s popiskem, který udává informace o letadlu jako jeho volací znak, směr letu letadla a podobně. V případě simulátoru je u popisku letadla zobrazováno jméno agenta, směr letu a stav, ve kterém se agent aktuálně nachází (viz Obrázek 1).

2.1 PROSTŘEDÍ

V agentních systémech prostředí reprezentuje prostor, ve kterém agenti existují a plní své cíle. Hlavní funkcí tohoto prostoru je detekce a vykonávání akcí, které jsou volány agenty v závislosti na vykonávaném agentním plánu [1]. V simulátoru jsou pro vykonávání těchto akcí implementovány metody zajišťující přepočet souřadnic polohy a směru letu agenta a tím pohyb agenta doprava/doleva o jeden stupeň nebo rovně o jeden pixel v závislosti na vykonávané akci.

Z důvodu rizika kolizí mezi letadly je v prostředí implementován mechanismus pro detekci těchto konfliktů. Kontrola rizika konfliktů je prováděna po vykonání každé akce agenta a je založena na porovnání polohy ostatních agentů vůči tomuto agentovi a určení, zda se ve výseči o úhlu 140° před kontrolovaným letadlem nachází nějaký narušitel. Výsledkem kontroly je vložení predikátu *current(normal)* (běžný let bez konfliktu) nebo *current(collision)* (detekováno riziko kolize) symbolizujícího stav, ve kterém se agent aktuálně nachází. Pro správné rozhodování je do báze znalostí agenta současně vložen predikát *previous(normal)* nebo *previous(collision)* určující agentův předchozí stav. Současně se v bázi agenta nachází pouze jeden predikát *current* a jeden predikát *previous*.

2.2 AGENT

Agent je reaktivní systém s určitou mírou vlastní inteligence, který je schopný na základě predikátů ve své bázi znalostí sám rozhodnout, jak nejlépe dosáhnout cíle, pro který je určen [1]. V mé implementaci je počátečním cílem každého agenta doletět do svého cílového bodu. Agent má v knihovně plánů k dispozici tři základní plány napsané v jazyce *AgentSpeak*, které pro dosažení cíle používá. Jedná se o plán pro let rovně o N pixelů a plán pro vykonání otočky doprava nebo doleva o α stupňů. V následující ukázce je zobrazen plán pro let rovně o N pixelů.

```
+!fly_straight(N) : N > 0
   <-
       fly(straight);
       -step(straight, N);
       +step(straight, N-1);
       ?check;
       !!fly.

+!fly_straight(N) <- -step(straight, N); internal.setNextStep; !!fly.
```

Při spuštění simulátoru se začne vykonávat plán pro dosažení cíle *!fly*, který je počátečním cílem každého agenta a symbolizuje let do koncového bodu. Výběr aplikovatelného plánu závisí na přítomnosti predikátu *step(typ,parametr)* v bázi znalostí agenta. Tento predikát určuje krok trasy a skládá se z *typu kroku* (straight/left/right) a *parametru* (celočíselná hodnota). Agent postupně vkládá do své báze znalostí tyto predikáty získané výpočtem optimální trasy ze startovního do cílového bodu na základě *Dubinsových křivek*. Popsaný princip výběru plánu je zobrazen v ukázce kódu níže.

```
+!fly : step(straight, N) <- !fly_straight(N).
+!fly : step(left, N) <- !turn_left(N).
+!fly : step(right, N) <- !turn_right(N).
+!fly <- ?check; !!fly.
```

3 ŘEŠENÍ KONFLIKTŮ

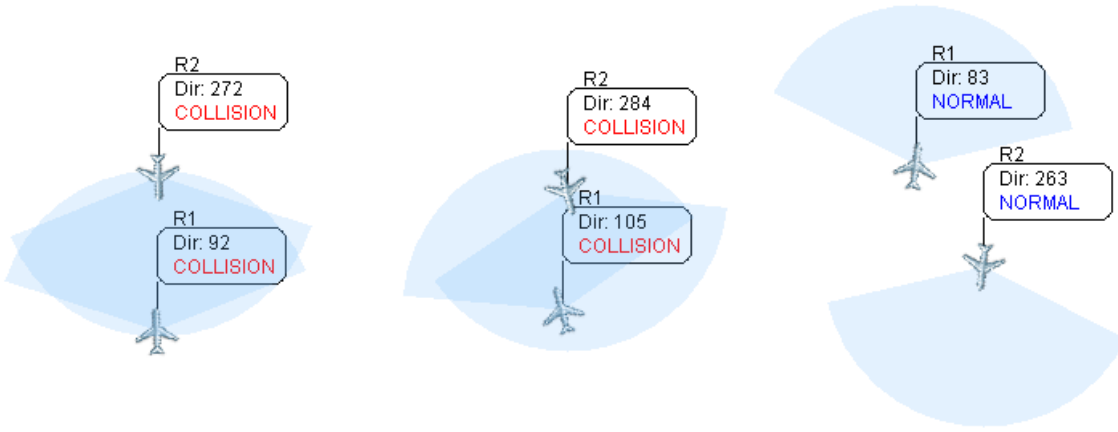
V rámci této práce jsou řešeny možné kolize mezi letadly. Agenti mohou tyto kolize řešit dvojím způsobem a to buď *vyhnutím se* nebo *pronásledováním*. Kterou reakci agent použije závisí na tom, zda detekovaný kolizní agent patří do přátelské nebo nepřátelské skupiny letadel a na dalších predikátech, které má v bázi znalostí.

Pro detekci kolize během vykonávání agentních plánů je vytvořen testovací cíl *?check*, který je průběžně prováděn. V kontextech plánů pro vykonání tohoto cíle jsou kombinace predikátů, na kterých závisí další průběh pohybu agenta.

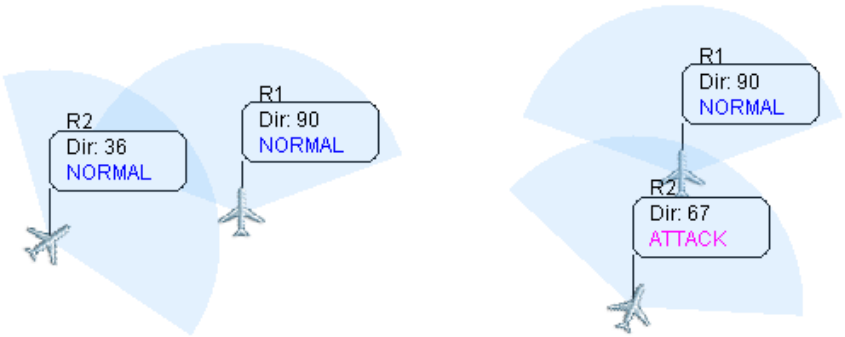
```
+?check : start & current(normal) <- ...
+?check : start & current(collision) <- ...
+?check : previous(normal) & current(collision) <- ...
+?check : previous(collision) & current(collision) <- ...
+?check : previous(collision) & current(normal) <- ...
+?check.
```

Pro detekci kolize v kontextu plánu musí být ve znalostní bázi agenta zároveň přítomen predikát *previous(normal)* (případně predikát *start*) a predikát *current(collision)* (viz kód výše). V takovém případě agent přeruší vykonávání aktuálního plánu a pomocí interní akce vloží do své báze znalostí predikát *step(left, 1)* určující úhybný manévr jako otočku doleva o jeden stupeň. Úhel, o který se letadlo vykonávající manévr otočí, tedy není předem daný, ale závisí na poloze a směru letu obou letadel a na době přítomnosti predikátu *current(collision)* v agentově bázi znalostí.

Na Obrázku 1 je princip řešení kolize *vyhnutím se* znázorněn ve třech krocích.



Obrázek 1: Ukázka detekce a řešení kolize letadel vyhnutím se (platí pro letadla ve stejné skupině).



Obrázek 2: Ukázka detekce kolize a následné pronásledování (vzájemní nepřátelé).

Na Obrázku 2 je znázorněna detekce kolize a následné pronásledování agenta v kolizi. Řešení kolize vzájemně nepřátelských letadel je odlišné, místo úhybného manévru je vypočítána optimální trasa z agentova aktuálního bodu do aktuálního bodu pronásledovaného agenta a je proveden jeden krok příslušného plánu založeného na predikátu *step(typ,parametr)*.

4 ZÁVĚR

V dosavadním postupu bylo implementováno prostředí a jednotliví agenti, tedy jejich plány a reakce na podněty z prostředí. Dále byl vytvořen mechanismus pro detekci konfliktů mezi agenty. Agenti jsou tedy schopni reagovat na vzniklé konflikty a dynamicky jim přizpůsobovat své plány pro dosažení cíle. V současné době pracuji na vytvoření uživatelského rozhraní, které umožní vybrat přednastavený model pro simulaci, vytvářet vlastní simulační modely a přehledně interpretovat vybranou simulaci.

REFERENCE

[1] Rafael H. Bordini.: *Programming multi-agent systems in AgentSpeak using Jason*. 2007, ISBN 978-0-470-02900-8.