Univerzita obrany
Fakulta vojenského leadershipu
Katedra taktiky

Softwarová dokumentace

Optimální přesun jednotek na bojišti

Optimal movement of units on the battlefield

Softwarová knihovna

Autoři: pplk. doc. Ing. Petr STODOLA, Ph.D.
npor. Ing. Jan NOHEL, Ph.D.

Brno 2018

Obsah

1. Popis knihovny

Předkládaná softwarová knihovna je výsledkem řešení dílčího cíle č. 1 dlouhodobého záměru rozvoje organizace PASVŘ II (Pokročilý automatizovaný systém velení a řízení II). Jedním z výstupů tohoto dílčího cíle bylo vyvinout softwarovou knihovnu pro optimalizaci osy přesunu jednotek na bojišti.

Mezi základní funkce, které knihovna nabízí, patří:

- Inicializace knihovny.
- Vkládání a úprava jednotek na bojišti.
- Nastavení meteorologických podmínek.
- Nastavení typu a parametrů přesouvající se jednotky.
- Výpočet optimální osy přesunu.

Softwarová knihovna se mimo jiné využívá jako součást vlastního taktického systému TDSS (Tactical Decision Support System), který umožňuje grafickou vizualizaci prostředí a je navržen v rámci systému PASVŘ pro podporu rozhodování velitelů jednotek.

Součástí této dokumentace je především popis jednotlivých funkcí, které knihovna nabízí, a také příručka pro implementaci knihovny v softwarových projektech.

2. Technické požadavky

Softwarová knihovna je vytvořena ve verzi pro operační systém Windows XP (nebo vyšší verze) a Android. Knihovna je vytvořena v programovacím jazyce C++ s využitím principů objektově orientovaného programování pod překladačem Microsoft Visual Studio.

Knihovna využívá funkce softwarové knihovny OTLibrary (Operačně-taktická softwarová knihovna). Bez knihovny OTLibrary nelze tuto knihovnu využívat. Podrobnosti ke knihovně OTLibrary jsou dostupné na https://vav.unob.cz/result/index/522853.

Pro správnou funkci knihovny je nutné mít k dispozici mapové podklady, které dodává Vojenský geografický a hydrometeorologický úřad v Dobrušce, konkrétně v následujících vrstvách:

- digitální model území 1:25 000 (DMU25),
- digitální model reliéfu DMR ve formátu BW3.

3. Popis funkcí knihovny

V této kapitole následuje přehled jednotlivých funkcí knihovny včetně stručného popisu. Kapitola je rozdělena do pěti tematických celků podle typů funkcí následovně:

- Inicializační funkce.
- Vkládání a úprava jednotek na bojišti.
- Nastavení meteorologických podmínek.
- Nastavení typu a parametrů přesouvající se jednotky.
- Výpočet optimální osy přesunu.

3.1 Inicializační funkce

Inicializační funkce slouží pro přípravu knihovny před jejím použitím. Inicializace knihovny je nutné realizovat prostřednictvím funkce:

```
bool Init(double utmx, double utmy, double w, double h);
```

Vstupními parametry je levý spodní rok prostoru bojiště v souřadnicovém systému WGS84/UTM (parametry `utmx`, `utmy`) a šířka a výška prostoru v metrech (parametry `w`, `h`). Návratová hodnota funkce je typu BOOL podle úspěšnosti (hodnota TRUE značí úspěšnou inicializaci knihovny, hodnota FALSE naopak neúspěch).

Po skončení práce s knihovnou je možné knihovnu uzavřít prostřednictvím funkce:

```
BOOL Close();
```

Tento krok není povinný. Pokud nebude uzavření voláno uživatelem knihovny, dojde k uzavření automaticky při ukončení programu. Návratová hodnota funkce je typu BOOL podle úspěšnosti.

Pro vyprázdnění paměti (např. před novou úlohou optimalizace trasy) lze využít funkci, která z paměti odstraní všechny dosud načtené nebo vložené objekty a nastavení:

```
void DeleteAll();
```

Pro načtení digitálních geografických dat je nutné využít funkcí knihovny OTLibrary, konkrétně `OTLibLoadDMR` a `OTLibLoadDMU`. Knihovnu je předtím nutné inicializovat funkcí `OTLibInit`. Podrobnosti jsou uvedeny v dokumentaci ke knihovně OTLibrary.

3.2 Vkládání a úprava jednotek na bojišti

Funkce slouží pro vkládání, úpravu a odstraňování jednotek rozmístěných na bojišti. Takto lze simulovat potřebnou taktickou situaci (jednotky na bojišti ovlivňují výpočet optimální trasy přesunu).

Knihovna definuje typ jednotek následovně:

```
enum UNITTYPE
{
	FRIENDLY,		// Vlastní jednotky
	HOSTILE,		// Nepřátelské jednotky
	NEUTRAL,		// Neutrální jednotky
	UNKNOWN			// Neznámé jednotky
};
```

Přidání jednotky se provede funkcí:

```
bool AddUnit(double utmx, double utmy, double r, UNITTYPE type);
```

Parametry `utmx` a `utmy` definují pozici jednotky v souřadnicovém systému WGS84/UTM, parametr `r` značí maximální dohled jednotky (tj. maximální vzdálenost pro výpočet viditelnosti) a parametrem `type` lze specifikovat typ jednotky. Návratová hodnota funkce je typu BOOL podle úspěšnosti.

Odstraňování již vložené jednotky (jednotek) lze provést funkcemi:

```
bool DeleteUnit(int idx);

void DeleteAllUnits();
```

Funkce `DeleteUnit` odstraní jednotku danou parametrem `idx`, který značí pořadí vložení jednotky (počínaje 0). Návratová hodnota funkce je typu BOOL podle úspěšnosti. Funkcí `DeleteAllUnits` dojde k odstranění všech vložených jednotek.

Funkce pro úpravu parametrů již vložené jednotky neexistují. V případě potřeby upravit parametry jednotky (např. pozice) je nutné ji nejprve odstranit a pak následně vložit znovu s novými parametry.

Zjištění parametrů již vložených jednotek se provádí funkcí:

```
bool GetUnit(int idx, double &utmx, double &utmy, double &r, UNITTYPE &type);
```

Parametr `idx` specifikuje index jednotky (pořadí vložení). Parametry `utmx`, `utmy`, `r`, `type` jsou výstupními parametry. V případě úspěšného volání funkce budou naplněny odpovídajícími hodnotami. Návratová hodnota funkce je typu BOOL podle úspěšnosti.

Celkový počet vložených jednotek na bojišti lze zjistit funkcí:

```
int GetNumUnits();
```

3.3 Nastavení meteorologických podmínek

Pro nastavení meteorologických podmínek, které ovlivňují optimální přesun jednotky, využívá knihovna dvě funkce. První funkce umožňuje měnit typ povrchu, po kterém se jednotka pohybuje, v závislosti na aktuálních meteorologických podmínkách. Knihovna definuje čtyři změny povrchů:

```
enum SURFTYPE
{
	NORMAL,			// Normální povrch
	SOAKED,			// Promáčený povrch
	SNOWCOVERED,		// Povrch pokryty sněhem
	FROZEN			// Zmrzlý povrch
};
```

Funkce pro změnu typu povrchu je následující:

```
bool SetMeteoSurface(SURFTYPE type, double amount=0.0);
```

Parametr `type` označuje způsob, jakým se mění povrch v důsledku působení meteorologických faktorů. U promáčeného povrchu (`SOAKED`) lze parametrem `amount` ovlivnit množství dešťových srážek (udává se v milimetrech). U sněhem pokrytého povrchu (`SNOWCOVERED`) lze parametrem

`amount` specifikovat výšku sněhové pokrývky (v centimetrech). U normálního povrchu (`NORMAL`) a zmrzlého povrchu (`FROZEN`) nemá hodnota parametru `amount` žádný význam. Návratová hodnota funkce je typu BOOL podle úspěšnosti.

Druhá funkce umožňuje plošně ovlivňovat viditelnost vlivem meteorologických podmínek. Při snížené viditelnosti lze procentuálně snižovat dohled jednotek. Návratová hodnota funkce je typu BOOL podle úspěšnosti.

```
bool SetMeteoVisibility(double vis);
```

Parametr `vis` udává procentuální změnu dohledu jednotek (v rozsahu 0 až 100). Hodnota 100 značí, že nedochází ke změně (platí hodnoty nastavené u jednotlivých jednotek), hodnotou 0 lze nastavit u všech jednotek nulový dohled.

3.4 Nastavení typu parametrů přesouvající se jednotky

Model umožňuje plánovat přesun pro tři základní typy elementů (prostředků):

```
enum VEHICLETYPE
{
	PERSON,				// Osoba
	WHEELED_VEHICLE,		// Kolové vozidlo
	TRACKED_VEHICLE		// Pasové vozidlo
};
```

Konkrétní typ elementu/vozidla se potom nastaví specifikací průměrných rychlostí, jakými se element je schopen přesunovat na různých typech povrchů za optimálních podmínek, tzn. na rovině, na povrchu neovlivněném meteorologickými podmínkami a bez přítomnosti nepřátelských jednotek. Typy povrchů jsou dány digitálním modelem území následovně:

```
enum SURFSPEED
{
	FIELD,				// Pole
	HIGHWAY,			// Dálnice
	CLASS_1ST,			// Silnice 1. třídy
	CLASS_2ND,			// Silnice 2. třídy
	CLASS_3RD,			// Silnice 3. třídy
	STREET,				// Ulice
	UNSURFACED,			// Nezpevněná cesta
	ROADOTHER,			// Ostatní cesty
	FOREST,				// Les
	PASTURE,			// Pastviny, louky
	FORESTOTHER,			// Ostatní typy vegetace
	WATER,				// Voda
	BUILDING			// Zastavěná oblast
};
```

Pro změnu typu elementu slouží funkce:

```
bool SetVehicleType(VEHICLETYPE type);
```

Parametr `type` značí typ elementu (`PERSON`, `WHEELED_VEHICLE`, `TRACKED_VEHICLE`). Návratová hodnota funkce je typu BOOL podle úspěšnosti.

Pro zjištění aktuálně nastaveného typu jednotky lze využít funkci:

```
VEHICLETYPE GetVehicleType();
```

Pro nastavení maximálních rychlostí elementu na různých typech povrchů za optimálních podmínek slouží funkce:

```
bool SetSurfaceSpeed(SURFSPEED surface, double speed);
```

Parametr `surface` určuje typ povrchu, parametr `speed` rychlost jednotky na tomto povrchu v kilometrech za hodinu. Návratová hodnota funkce je typu BOOL podle úspěšnosti. Při potřebě zamezit/zakázat pohyb jednotky na určitém typu povrchu lze nastavit pro tento typ rychlost 0.

Pro zjištění aktuálně nastavené rychlosti elementu na různých typech povrchů slouží funkce:

```
double GetSurfaceSpeed(SURFSPEED surface);
```

Další dvě funkce slouží pro uložení/načtení kompletního souboru hodnot rychlostí elementu do/ze souboru. Lze takto vytvořit soubory se šablonami pro nejrůznější konkrétní typy prostředků. Parametrem obou funkcí je specifikace jména souboru (`filename`), do/ze kterého budou data ukládána/načítána. Návratová hodnota obou funkcí je typu BOOL podle úspěšnosti.

```
bool SaveSurfaceSpeed(char *filename);

bool LoadSurfaceSpeed(char *filename);
```

Formát souboru pro ukládání rychlostí prostředku je uveden níže (hodnoty jsou ilustrační). Typ může nabývat hodnot `Person` (osoba), `Wheeled` (kolové vozidlo) nebo `Tracked` (pásové vozidlo). Hodnoty rychlostí jsou uváděny v kilometrech za hodinu.

```
Type = Wheeled
Speed_Highway = 100.00
Speed_1stClass = 90.00
Speed_2ndClass = 80.00
Speed_3rdClass = 70.00
Speed_Street = 50.00
Speed_Unsurfaced = 25.00
Speed_RoadOther = 60.00
Speed_Forest = 0.00
Speed_Pasture = 0.00
Speed_Field = 10.00
Speed_ForestOther = 0.00
Speed_Water = 0.00
Speed_Build = 0.00
```

3.5 **Výpočet optimální osy přesunu**

Po nastavení všech potřebných parametrů pro přesun je možno přistoupit k výpočtu optimální osy přesunu jednotky. To se realizuje funkcí:

```
bool ComputePath(double utmx1, double utmy1,
                 double utmx2, double utmy2, double step);
```

Parametry `utmx1` a `utmy1` určují výchozí polohu jednotky v prostoru. Parametry `utmx2` a `utmy2` určují požadovanou cílovou polohu jednotky v prostoru. Hodnoty jsou udávány v souřadnicovém systému

WGS84/UTM. Parametrem `step` se nastaví rasterizační krok, tzn. velikost čtverce, na které se celý prostor rozdělí. Výpočet optimální polohy pak probíhá v rámci takto rasterizovaného prostoru. Velikost rasterizačního kroku lze nastavit libovolně s podmínkou, že celkový počet čtverců nepřesáhne hodnotu 4000 × 4000. Návratová hodnota funkce je typu BOOL podle úspěšnosti, tzn. v případě úspěšného nalezení osy přesunu bude návratová hodnota TRUE.

V případě úspěšného nalezení osy přesunu lze následující funkcí zjistit počet bodů, ze kterých je složena optimální trasa (vzdálenost jednotlivých bodů je dána velikostí rasterizačního kroku):

```
int GetPathNumPoints();
```

Další funkcí lze potom procházet jednotlivé body:

```
bool GetPathPoint(int idx, double &utmx, double &utmy);
```

Parametr `idx` značí index (pořadí) bodu (počínaje 0), ve výstupních parametrech `utmx1` a `utmy1` jsou v případě úspěšného volání funkce uloženy souřadnice vybraného bodu osy přesunu v souřadnicovém systému WGS84/UTM. Návratová hodnota funkce je typu BOOL podle úspěšnosti.

Následujícími funkcemi lze zjistit celkovou délku osy přesunu (v metrech) a odhadovaný čas přesunu jednotky z výchozího do cílového bodu (v sekundách):

```
double GetPathDist();

double GetPathTime();
```

4. Implementace knihovny do softwarového projektu

Následující kapitola je stručnou příručkou pro implementaci knihovny ve vlastních softwarových projektech. Knihovna je nativně určena pro projekty v jazyce C++ ve vývojovém prostředí Visual Studio.

Ve verzi pro operační systém Windows se knihovna skládá ze čtyř souborů:

- OTModels.h
- OTModelsNavig.h
- OTModels.lib
- OTModels.dll

Ve verzi pro operační systém Android se knihovna skládá ze tří souborů:

- OTModels.h
- OTModelsNavig.h
- libOTModels.so

V souborech OTModels.h a OTModelsNavig.h jsou deklarovány všechny funkce uvedené v kapitole 3. Tyto soubory se vkládají do vlastního projektu. Soubory OTModels.lib a OTModels.dll tvoří vlastní programovou realizaci uvedených funkcí pro operační systém Windows, soubor libOTModels.so tvoří vlastní programovou realizaci pro operační systém Android.

Pro správnou funkci knihovny je nezbytná přítomnost operačně-taktické knihovny OTLibrary. Funkce a možnosti použití této knihovny jsou uvedeny v samostatné dokumentaci. Z této knihovny se využívají především funkce pro načtení digitálních geografických dat (digitální model území, digitální model reliéfu), ale lze využít i další funkce, např. pro převod souřadnic mezi souřadnicovými systémy. Tato knihovna je k dispozici pro operační systém Windows a Android a skládá se z následujících souborů:

- OTLibrary.h
- OTLibrary.cfg
- OTLibrary.lib (pro Windows)
- OTLibrary.dll (pro Windows)
- OTLibrary.so (pro Android)

Základní postup při využívání jednotlivých funkcí knihovny je popsán v následující části. První fáze je součástí funkcí knihovny OTLibrary, ostatní fáze jsou součástí knihovny OTModels.

- Inicializace operačně taktické knihovny a načtení digitálních geografických dat.
- Inicializace knihovny pro výpočet optimální osy přesunu.
- Nastavení parametrů scénáře (vložení jednotek na bojišti, nastavení meteorologických podmínek, nastavení parametrů přesouvající se jednotky).
- Výpočet optimální osy přesunu.
- Procházení jednotlivých bodů osy přesunu, zjištění odhadované délky a doby přesunu.