UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY
A INFORMATIKY

SIETE SENZOROV PRE MONITOROVANIE ROVINY POUŽITÍM ROTAČNÝCH A LÚČOVÝCH SENZOROV

Diplomová práca

2013

Bc. Marek Manduch

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY
A INFORMATIKY

SIETE SENZOROV PRE MONITOROVANIE ROVINY POUŽITÍM ROTAČNÝCH A LÚČOVÝCH SENZOROV

Diplomová práca

Študijný program: informatika
Študijný odbor: 2508 informatika
Školiace pracovisko: Katedra informatiky
Školiteľ: Mgr. Stefan Dobrev, PhD.

Bratislava, 2013

Bc. Marek Manduch

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Marek Manduch
Študijný program: informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský

Názov: Siete senzorov pre monitorovanie roviny použitím rotačných a lúčových senzorov.

Cieľ: Ciele diplomovej práce:
1) návrh a implementácia programu na návrh, vizualizáciu a overovanie monitorovacích senzorických sietí používajúcich rotujúce a čiarové antény
2) použitie tohoto programu a jeho rozšírení na systematický prieskum priestoru možných riešení (konfigurácií senzorov) pre rôzne modelové varianty
3) teoretický návrh a analýza monitorovacích senzorových sietí

Vedúci: RNDr. Štefan Dobrev, PhD.
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: doc. RNDr. Daniel Olejár, PhD.

Dátum zadania: 30.11.2011

Dátum schválenia: 30.11.2011 prof. RNDr. Branislav Rovan, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Čestne prehlasujem, že som túto diplomovú prácu vypracoval samostatne s použitím citovaných zdrojov.

Bratislava _____

Chcel by som poďakovať môjmu školiteľovi za množstvo hodín, ktoré sa mi venoval a pomoc pri vytváraní tejto práce.

Abstrakt

Táto diplomová práca sa venuje návrhu a implementácii algoritmu pre overovanie správnosti návrhu siete senzorov pre monitorovanie roviny a jednoduchých mnohouholníkov použitím rotačných a lúčových senzorov. Vytvárame formálny popis použitého modelu, ktorý pre účely automatického overovania rozširujeme o nové pojmy (význačný bod, sektor, oblasť, ...) podrobnejšie popisujúce konfiguráciu systému. Definujeme pojmy, ktoré nám umožňujú sledovať dynamické správanie systému a vyhodnotiť, v ktorých oblastiach sa môže votrelec nachádzať a v ktorých nie.

Kľúčové slová: detekcia votrelca, monitorovanie oblasti, sieť senzorov, rotačný senzor, lúčový senzor, dynamické monitorovanie

Abstract

This thesis is devoted to the proposition and the implementation of the algorithm for the verification of the network sensor design for monitoring plane and simple polygons using rotating and beam sensors. We create the formal description of the used model. We extend this model for the intention of automatic verification with new concepts (significant point, sector, area, ...), which describe configuration of system in more detail. We define new concepts, which enable us to observe dynamic behavior of system and also enable us to determine in which areas can be an intruder and in which cannot be any intruder.

Keywords: intruder detection, area monitoring, sensor network, rotating sensor, beam sensor, dynamic monitoring

Obsah

Zoznam obrázkov

Úvod

Monitorovanie oblasti (polia, okolie sídla, priemyselné a vojenské areály, múzeá, výstavy) pre prípad vniknutia votrelca (zviera, zlodej, nepriateľský nájazd) alebo nebezpečnej udalosti (požiar) je staré ako ľudstvo samé.

Rozvoj techniky nahradil použitie ľudských a zvieracích zmyslov sofistikovanými senzormi, čo umožnilo podstatne rozsiahlejšie nasadenie monitorovania. Na druhej strane rozvoj spoločnosti a techniky tiež podstatne zvýšil požiadavky na monitorovanie. Toho, čo treba chrániť je omnoho viac, a aj schopnosti votrelca sú technikou posilnené, napr. teraz treba monitorovať aj vzdušný priestor.

Prirodzenou požiadavkou pri monitorovaní oblasti je minimalizácia nákladov a to vedie k minimalizácii počtu a schopností senzorov. Táto požiadavka spôsobuje, že pri návrhu rozmiestnenia senzorov sa častejšie vyskytujú chyby (slepé uhly a pod.), ktoré je ľahké urobiť najmä pri návrhu dynamických monitorovacích systémov, pri ktorých sa oblasť monitorovaná daným senzorom mení podľa toho ako sa senzor pohybuje/rotuje. Preto sa často riešia otázky, ako navrhovať siete senzorov a ako overovať správnosť týchto návrhov.

Táto problematika sa rozoberá v mnohých článkoch. Pri detekcii votrelca sa literatúra delí do dvoch skupín. V prvej skupine je sledované prekročenia hranice votrelcom a v druhej skupine je sieť senzorov používaná na sledovanie celej oblasti.

Vo výpočtovej geometrii sa tejto problematiky venujú rôzne témy zahŕňajúce oba prístupy. Sledovaniu prekročenia hranice monitorovanej oblasti sa venuje problém s názvom „Multi-agent patrolling problem“, ([6, 7]). S monitorovanou oblasťou sa pracuje, ako s grafom a po jeho hranách chodia tzv. „agenti“, ktorý majú navštíviť každý vrchol, tak aby minimalizovali čas, za ktorý sa im to podarí. Sledovaniu prekročenia hranice sa venuje aj množstvo iných článkov, v ktorých sú použité rôzne metódy [11, 12, 13, 14, 15, 18]).

V našej práci sa budeme zaoberať detekciou votrelca v celej monitorovanej oblasti. Tomuto prístupu sa vo výpočtovej geometrii venujú problémy ako „Art gallery problem“ ([8]) a „Guarding polyhedral terrains“ ([9]). Pri prvom zo spomenutých sa jedná o rozmiestnenie a použitie minimálneho počtu senzorov, ktorých zorný uhol je 360 stupňov a ich dosah je obmedzený len stenami. Druhý spomenutý problém pracuje s pojmami ako „edge guard“ a

„vertex guard“. Tieto dva druhy „strážcov“ pozorujú všetko, k čomu od ich niektorého bodu vedie úsečka nepretínajúca žiadnu stenu (hranu). Sledovaniu celej oblasti sa venujú aj články ako napríklad ([16, 19]).

Ďalšie rozdelenie typov sledovania oblasti je podľa toho, či sa jedná o statické sledovanie, alebo dynamické. Pri statickom sledovaní sa v čase nemení poloha senzorov ani oblasť, ktorú sledujú. Sieť senzorov buď monitoruje, alebo nemonitoruje oblasť. Pri dynamickom sledovaní sa senzory pohybujú/rotujú a oblasť sledovaná daným senzorom sa v čase mení. V tomto prípade berieme do úvahy aj čas, za ktorý sieť senzorov oblasť monitoruje, alebo aké je najväčšie možné oneskorenie detekcie votrelca po jeho vniknutí. My pracujeme s dynamickým sledovaním a hovoríme, že sieť senzorov monitoruje oblasť, ak môže odhaliť každého votrelca s konečným oneskorením po jeho príchode.

K sledovaniu možno používať rôzne typy senzorov. V našej práci uvažujeme dva základné typy. Jedným je lúčový (statický) senzor, pozostávajúci z páru zariadení: zdroj lúča a prijímač. Zdroj lúča vysiela lúč (svetlo, zvuk, atď.) priamo do prijímača. Tento typ senzora spozoruje votrelca, keď zablokuje lúč pretnutím úsečky medzi zdrojom a prijímačom.

Druhý typ senzora, ktorý používame, je rotačný. Tento senzor abstrahuje napríklad infračervené senzory, radary, alebo senzory založené na spracovaní obrazu kamery. Vďaka nim je možné detekovať votrelca v kruhovej oblasti. V danom čase pokrývajú úsek kruhovej oblasti a rotujú okolo pevného bodu. Takéto senzory boli použité napríklad aj v presnom 3D modelovaní scény [20].

Výsledky ohľadne monitorovania Euklidovskej roviny rotačnými a lúčovými senzormi poskytuje článok [1]. Intuícia získaná z navrhovania siete pre nekonečnú roviny by mohla byť užitočná pre návrh siete pre konečnú oblasť, avšak navrhovanie siete senzorov bez pomocných nástrojov nie jednoduché a návrh siete senzorov je potrebné overiť dôkazmi. Vytvoriť tieto dôkazy je časovo náročné a nástroje na dokazovanie správnosti návrhu neexistujú. Pre dokazovanie správnosti návrhu je dôležitá aj vizualizácia siete senzorov, ktorá pomáha pri hľadaní chýb v návrhu a ich odstraňovaní.

V tejto práci sa venujeme vytvoreniu a implementácii algoritmu na dokazovanie správnosti návrhu siete senzorov a jej vizualizácii. Pri vizualizácii sú zobrazované aj podstatné momenty dôkazu správnosti. Naším cieľom je vďaka tomuto algoritmu poskytnúť nástroj, vďaka ktorému sa uľahčí a urýchli riešenie problému sledovania monitorovanej oblasti.

Naša práca pozostáva z dvoch kapitol a prílohy. Prvá kapitola sa venuje teoretickým základom. Ako prvé formálne popíšeme základný model, podľa neformálneho modelu z [1]. Základný model rozširujeme pre účely automatického overovanie o nové pojmy (význačný bod, sektor, oblasť, ...) podrobnejšie popisujúce konfiguráciu systému. Definujeme pojmy, ktoré nám umožňujú sledovať dynamické správanie systému a vyhodnotiť, v ktorých oblastiach sa môže votrelec nachádzať a v ktorých nie. Druhá kapitola sa venuje návrhu a implementácii algoritmu automatického overenia. Popisujeme jednotlivé kroky algoritmu a netriviálne problémy, ktoré sa pri implementácii vyskytli. Príloha obsahuje aplikáciu, ktorá poskytuje vizualizáciu siete senzorov a overovanie jej návrhu. Príloha zahŕňa aj zdrojový kód, stručnú dokumentáciu, používateľskú príručku a príklady návrhov siete senzorov pre vyskúšanie aplikácie.

Kapitola 1

Teoretické základy

V tejto kapitole sa venujeme formálnemu popisu siete senzorov a pojmov, ktoré využívame k vysloveniu tvrdení. Tieto tvrdenia nám pomáhajú pri návrhu algoritmu pre overovanie siete senzorov a jeho implementácii.

1.1 Základný model

Formálne popíšeme model s ktorým pracujeme. Základný model a jeho definície sú vytvorené na základe modelu popísaného v článku [1]. Niektoré z nich sú doplnené a presnejšie popísané, či nahradené ekvivalentnými definíciami, ktoré v práci častejšie používame.

Definícia 1.1.1 (**monitorovaná oblasť** [1])**.** Monitorovaná oblasť M je buď nekonečná dvojrozmerná Euklidovská rovina, teda $M \equiv R^2$, alebo M je *jednoduchý mnohouholník* [2].

Votrelca, ktorého definujeme si môžete predstaviť, ako zlodeja, ktorý sa môže objaviť na ľubovoľnom mieste monitorovanej oblasti a v ľubovoľnom čase. No ak sa už raz objaví, môže sa pohybovať len po súvislej trajektórii (nepovoľujeme skoky) a len v monitorovanej oblasti. Ak by sme povolili, že môže odísť a vrátiť sa, problém jeho odhalenia by spočíval v tom, že by v monitorovanej oblasti nemohol zotrvať dostatočne dlho bez spozorovania. V konečnom dôsledku by sme riešili ekvivalentný problém.

Definícia 1.1.2 (**poloha votrelca** [1])**.** Poloha votrelca je definovaná funkciou $f_I(t) \rightarrow M \cup \perp$. Funkcia $f_I(t)$ spĺňa navyše podmienku, že $\forall t \in \mathbb{R}$ platí:

$(f_I(t)) \neq \perp) \Rightarrow ((\forall t' > t : f_I(t') \neq \perp) \wedge (f_I$ je spojitá na def. obore $\langle t, \infty)))$.

Pre spozorovanie votrelca používame dva druhy senzorov. Jeden je rotačný, ktorým je úsečka ktorá rotuje okolo svojho koncového bodu, buď v smere hodinových ručičiek, alebo v protismere hodinových ručičiek. Druhým typom je lúčový senzor (v pôvodnom texte "beam sensor"[1]). Ten je vlastne nepohyblivá statická úsečka.

Definícia 1.1.3 (**rotačný senzor** [1])**.** Rotačný senzor je objekt zadaný šesticou $(x, p, r, \Theta, I_p, d)$.

- x - centrum, alebo stred senzora, zadaný súradnicami.
- p - perióda rotácie (čas, za ktorý sa senzor dostane do pôvodnej pozície).
- r - dosah senzora (polomer kružnice, ktorú opisuje senzor počas rotácie).
- Θ - uhol snímania senzora
- I_p - inicializačná poloha senzora. (Veľkosť uhla, o ktorý je senzor otočený v smere hodinových ručičiek. Ak I=0 je súbežný s osou x.)
- d - smer otáčania senzora (*v smere hodinvoých ručičiek* alebo *v protismere hodinových ručičiek*)

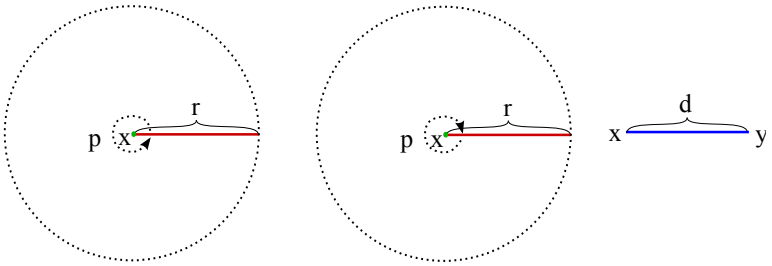
Poloha senzora s v čase t je určená zobrazením $D_s(t)$, ktoré pre čas t (na základe uvedených parametrov senzora s) vracia množinu bodov pokrytých senzorom s.

Definícia 1.1.4 (**lúčový senzor** [1])**.** Lúčový senzor s je objekt zadaný dvomi bodmi x a y, $x \in M \wedge y \in M$. Bodmi x a y je určená jeho dĺžka r. Funkcia určujúca polohu s v čase t je konštantná, teda $\forall t : D_s(t) = \overline{xy}$. x je začiatočný bod a y je koncový bod senzora s v čase t.

Definícia 1.1.5 (**sieť senzorov** [1])**.** Sieť senzorov S je množina, ktorej prvky sú rotačné a lúčové senzory.

Definícia 1.1.6 (**uniformná sieť senzorov** [1])**.** Uniformná sieť senzorov S je sieť senzorov, pre ktorú platí, že dĺžka všetkých rotačných senzorov r je rovnaká a každý lúčový senzor je dlhý nanajvýš r.

Poznámka 1.1.1. Ďalej v našej práci uvažujeme už len *uniformnú sieť senzorov* S, pre ktorú navyše platí, že všetky rotačné senzory majú uhol snímania $\Theta = 0$. A preto nám na vyjadrenie rotačného senzor postačí namiesto šestice $(x, p, r, \Theta, I_p, d)$, používať trojicu (x, y, d), kde x je začiatočný a y koncový bod v čase 0 rotačného senzora a d smer otáčania.



Obr. 1.1: Príklad rotačných senzorov otáčajúcich sa v smere a protismere hodinových ručičiek s periódou p, dosahom r, $\Theta = 0$ umiestnený v bode x, a lúčový senzor dĺžky $d \leq r$ medzi x and y.

Prvky siete senzorov slúžia k odhaleniu votrelca v monitorovanej oblasti, ktoré je uskutočnené nižšie definovanou udalosťou odhalenia votrelca.

Definícia 1.1.7 (**odhalenia votrelca** [1])**.** Odhalenia votrelca nastáva v čase t, ak platí, že $\exists s; s \in S : D_s(t) \cap f_I(t) \neq \emptyset$.

1.2 Zavedenie nových pojmov a tvrdení

K základnému modelu, s ktorým pracujeme pridávame nové pojmy. Pretože sieť senzorov analyzujeme v jednotlivých význačných časoch, v ktorých sa niečo udialo (stretnutie, rozídenie senzorov, či prekrytie senzorov), definujeme udalosti. Zaujímavé pre nás sú len tie časy, v ktorých nastáva niektorá z udalostí a v týchto časoch sú pre nás dôležité body ležiace na senzoroch (začiatky a konce senzorov, ich prieniky a konce prekrytí), ktoré nazývame význačné body.

1.2.1 Udalosti

Udalosti stretnutia a rozídenia senzorov presnejšie opisujú to, čo si pod nimi aj intuitívne môžeme predstaviť. Ak senzory nemali spoločný bod a následne majú, potom sa jedná o udalosť stretnutia (konkrétne pretnutia). Ak senzory mali spoločný bod a následne nemajú potom ide o rozídenie senzorov. Špeciálne ak senzory majú v istom čase spoločných viac bodov ako jeden a predtým mali najviac jeden spoločný bod, potom ide o udalosť prekrytia.

Definícia 1.2.1 (**udalosť stretnutia senzorov**)**.** Udalosť stretnutia nastáva v čase t, keď pre senzory s_1, s_2, ... , s_k, kde $k > 1$, a pre množiny bodov A a B platí nasledovné:

$$D_{s_1}(t) \cap D_{s_2}(t) \cap ... \cap D_{s_k}(t) = A$$

a zároveň existuje $\varepsilon > 0$, také že:

$$\forall t', (t - \varepsilon) < t' < t : D_{s_1}(t') \cap D_{s_2}(t') \cap ... \cap D_{s_k}(t') = B$$

Ak $|A| > 1$ a $|B| \leq 1$ hovoríme aj, že nastala **udalosť prekrytia**.
Ak $|A| = 1$ a $|B| = 0$ hovoríme aj, že nastala **udalosť pretnutia**.
Množina A určuje miesto, v ktorom udalosť nastala.

Definícia 1.2.2 (**udalosť rozídenia senzorov**)**.** Udalosť rozídenia nastáva v čase t, keď pre senzory s_1, s_2, ... , s_k, kde $k > 1$, a pre množinu bodov A platí nasledovné:

$$D_{s_1}(t) \cap D_{s_2}(t) \cap ... \cap D_{s_k}(t) = A; |A| = 1$$

a zároveň existuje $\varepsilon > 0$, také že:

$$\forall t', t < t' < (t + \varepsilon) : D_{s_1}(t') \cap D_{s_2}(t') \cap ... \cap D_{s_k}(t') = \emptyset$$

Množina A určuje miesto, v ktorom udalosť nastala.

Sieť senzorov chceme skúmať v diskrétnych hodnotách času. Najdôležitejšie pre nás sú časy, v ktorých nastáva aspoň jedna z udalostí stretnutia alebo rozídenia. Pre jednoduchšie pohybovanie sa po týchto hodnotách zavádzame pojmy predchádzajúci a nasledujúci krok.

Definícia 1.2.3 (**predchádzajúci krok k času** t)**.** Predchádzajúci krok k času t je čas $t_B = max\{t^{'}|t^{'} < t \wedge t^{'}$ je čas udalosti stretnutia, alebo rozídenia$\}$.

Definícia 1.2.4 (**nasledujúci krok k času** t))**.** Nasledujúci krok k času t je čas $t_N = min\{t^{'}|t^{'} > t \wedge t^{'}$ je čas udalosti stretnutia, alebo rozídenia$\}$.

1.2.2 Význačné body a oblasti

Nasleduje zavedenie nových pojmov, ako význačný bod, sektor a oblasť vďaka ktorým sa dá ľahšie zachytiť, ako sieť senzorov vyzerá v rôznych časoch a hľadať súvislosti medzi jednotlivými natočeniami siete.

Význačný bod je bod ležiaci na jednom zo senzorov, pričom je buď koncovým bodom senzoru, alebo je prienikom viacerých senzorov. Nasleduje formálna definícia.

Definícia 1.2.5 (**význačný bod**)**.** Význačný bod je dvojica (x, t), kde $x \in M$ a platí jedna z možností:

- $\exists s \in S$, že x je koncový alebo začiatočný bod s v čase t.
- x je prienikom aspoň dvoch senzorov v čase t.

Množina $P_{S,t}$ je označením pre **množinu všetkých význačných bodov** v čase t pre sieť senzorov S.

Každý význačný bod (x, t) má aspoň jednu z nasledovných vlastností:

- *hraničný bod senzoru*: $\exists s \in S$, že x je koncový alebo začiatočný bod s v čase t.
- *priesečník*: x je prienikom aspoň dvoch senzorov v čase t.
- *hraničný bod prekrytia*: x je začiatočný, alebo koncový bod prekrytia dvojice senzorov v čase t.

Na obrázku 1.2 a) sú dva senzory s_1 a s_2. V čase t_1 vytvárajú päť význačných bodov $p_1, ..., p_5$. Vlastnosť *hraničný bod senzoru* majú význačné body $p_1, ..., p_4$ a p_5 má vlastnosť *priesečník*. Na obrázku 1.2 b) sú význačné body $p_6, ..., p_9$, kde p_7 a p_8 majú vlastnosť *hraničný bod senzoru* a zároveň vlastnosť *hraničný bod prekrytia*.

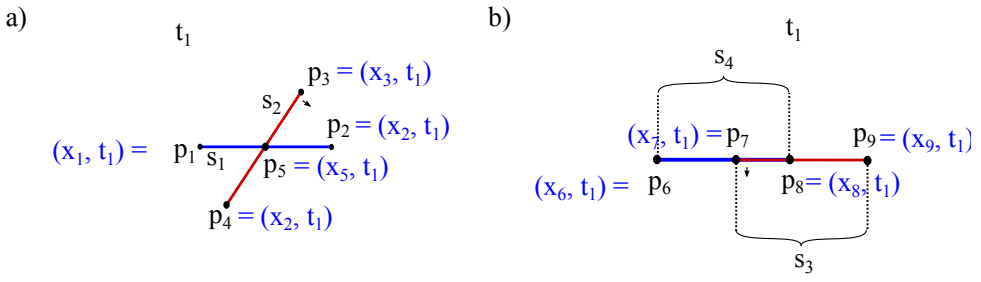
Často pracujeme s úsečkami, ktoré celé ležia na senzore v čase t a ich koncami sú význačné body. Pokiaľ medzi koncami tejto úsečky neležia žiadne význačné body, nazývame ich *sektory*. Nasleduje formálna definícia.

Definícia 1.2.6 (sektor). Sektor je dvojica význačných bodov $((x,t),(y,t))$, pričom platí, že $\exists s \in S : x, y \in D_s(t)$ a zároveň $\forall (z,t); z \in D_s(t), z \neq x, z \neq y$ platí, že z neleží na úsečke $\overline{xy}$.

Definícia 1.2.7 (indukovaný graf pre sieť senzorov v čase t). K sieti senzorov S v čase t vytvoríme planárny graf $G_{S,t}$ nasledovne: vrcholmi sú význačné body a pre každý sektor (p_1, p_2) pridáme hranu medzi vrcholy reprezentujúce význačné body p_1 a p_2. Takto zostrojený graf $G_{S,t}$ nazývame indukovaný graf pre sieť senzorov S v čase t.

Ak z každého komponentu grafu $G_{S,t}$ vytvoríme nový graf, potom steny týchto nových grafov nazývame **jednoduché oblasti**. Jednoduchá oblasť pokrýva množinu bodov. Funkcia $S(O)$ vracia množinu bodov, ktorú pokrýva jednoduchá oblasť O. Jednoduchá oblasť O je určená dvojicou (V,t), kde (V) je zoznam význačných bodov určujúci danú stenu a t je čas, pre ktorý sme zo siete senzorov zostrojili graf $G_{S,t}$.

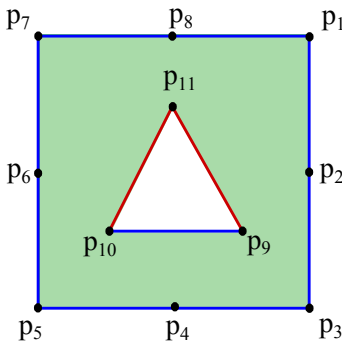
Steny grafu sú dvoch druhov – práve jedna stena (vrámci každého grafu zostrojeného z komponentu $G_{S,t}$) je neohraničená, a nazýva sa vonkajšia stena. Ostatné steny sa nazývajú vnútorné. Preto aj jednoduchá oblasť je buď vonkajšia alebo vnútorná (podľa príslušnej steny).



Obr. 1.2: Znázornenie význačných bodov tvorených senzormi s_1, s_2, s_3 as_4.

Definícia 1.2.8 (oblasť). Oblasť O pre čas t je trojica (Out, In, t) popisujúcu množinu bodov $S(O)$ takú, že pre každé dva body u,v, z A existuje spojitá krivka K spájajúca u s v taká, že K nepretína žiadny senzor z S. Pričom

- Out - je množina popisujúca vonkajšiu hranicu. (Out, t) tvorí jednoduchú oblasť (vnútornú).
- In - množina množín význačných bodov opisujúcich vnútorné výseky. $\forall F \in In$ platí, že (F, t) je jednoduchá oblasť (vonkajšia) a $\forall F_1, F_2 \in In; F_1 \neq F_2 : S((F_1, t)) \cap S((F_2, t)) = \emptyset$, že S
- t - čas v ktorom uvažujeme danú oblasť.
- $S(O)$ - je množina, ktorú oblasť O pokrýva; $S(Out, t) - \bigcup_{F \in In} S((F, t))$



Obr. 1.3: Príklad oblasti $O = (Out, In, t)$, kde $Out = \{p_1, ..., p_8\}$, $In = \{\{p_9, ..., p_{11}\}\}$ a $S(O)$ je množina bodov zvýraznená zelenou farbou.

Je potrebné zachytiť kam sa ktorý bod pohne po otočení siete senzorov. Intuícia za ekvivalenciou posunutia, ktorú zavádzame je nasledovná: ak otočíme rotačný senzor, tak význačné body opisujúce jeho koniec pred a po otočení opisujú ten istý objekt (stále je to ten istý koniec senzoru), ale v rôznych uhloch otočenia (resp. v rôznych časoch) a teda tieto význačné body sú ekvivalentné ekvivalenciou posunutia. Alebo ak majú dva senzory priesečník, otočíme ich a pritom počas otáčania mali stále priesečník (nenastala medzi nimi udalosť rozídenia), tak význačný bod opisujúci ich priesečník pred a po otočení opisujú ten istý priesečník, ktorý sa len niekam mohol posunúť v dôsledku otočenia senzorov.

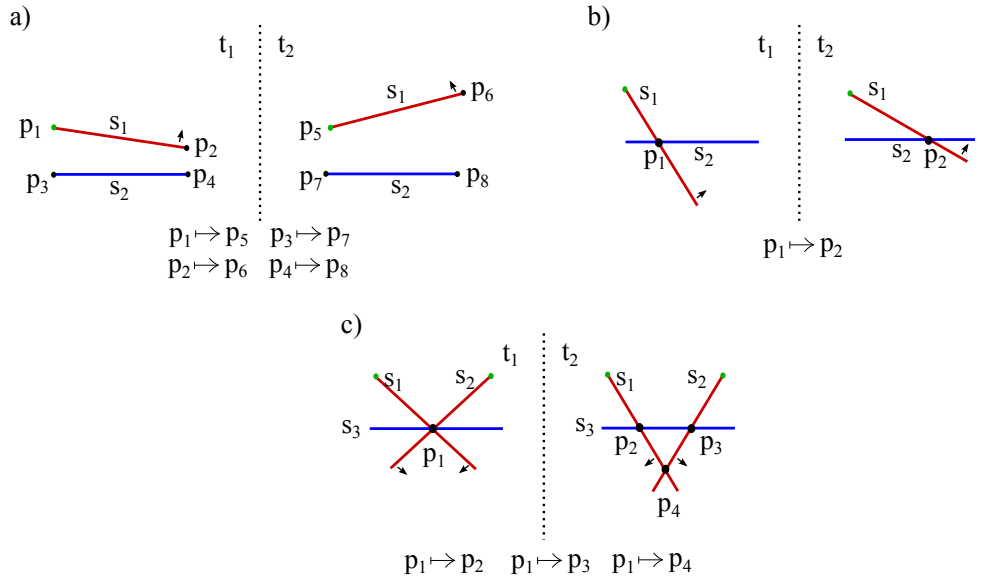
Zadefinujeme formálne ekvivalenciu posunutia medzi význačnými bodmi. Táto ekvivalencia je popísaná dvomi možnosťami. Buď ide o posunutie koncového bodu senzoru, alebo o posunutie priesečníka senzorov.

V druhej podmienke ekvivalencie posunutia význačných bodov (x_1, t_1) a (x_2, t_2) popisujeme nutnosť existencie takej dvojice senzorov, že x_1 a x_2 sú ich priesečníky v časoch t_1 a t_2 a medzi časmi t_1 a t_2 sa táto dvojica senzorov nerozišla.

Definícia 1.2.9 (ekvivalencia posunutia význačných bodov $\mapsto$). Hovoríme, že význačné body (x_1, t_1) a (x_2, t_2) sú ekvivalentné v rámci ekvivalencie posunutia a zapisovať $(x_1, t_1) \mapsto (x_2, t_2)$, ak platí jedna z nasledujúcich podmienok:

- bod x_1 je v čase t_1 koncový (resp. začiatočný) bod senzora $s \in S$ a x_2 je v čase t_2 koncový (resp. začiatočný) bod toho istého senzora.

- ak pre (x_1, t_1) platí : $x_1 \in D_{s_1}(t_1) \cap D_{s_2}(t_1) \cap ... \cap D_{s_n}(t_1)$, a zároveň pre (x_2, t_2) platí : $x_2 \in D_{s_1'}(t_2) \cap D_{s_2'}(t_2) \cap ... \cap D_{s_m'}(t_2)$. Potom ak $\exists s_i, s_j : s_i \neq s_j \wedge s_i \in \{s_1, ..., s_n\} \cap \{s_1', ..., s_n'\} \wedge s_j \in \{s_1, ..., s_n\} \cap \{s_1', ..., s_n'\}$ a zároveň neexistuje udalosť, ktorá nastala medzi časmi t_1 a t_2 a $\{s_i, s_j\}$ je podmnožina množiny jej aktérov.



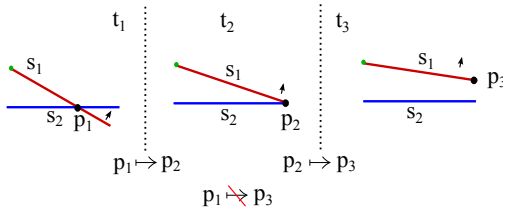
Obr. 1.4: Príklad ekvivalencií význačných bodov.

Poznámka 1.2.1. Je dôležité si uvedomiť, že jeden význačný bod v čase t_1 môže byť ekvivalentný s viacerými význačnými bodmi s časmi t_2, kde $t_1 \neq t_2$. Ak $t_1 > t_2$ hovoríme o **zlúčení význačných bodov**. Ak $t_2 > t_1$ hovoríme o **rozmnožení význačných bodov** (viď obrázok 1.4 c)).

Na obrázku 1.4a) ekvivalencie $p_1 \mapsto p_5$, $p_2 \mapsto p_6$, $p_3 \mapsto p_7$ a $p_4 \mapsto p_8$ platia podľa prvej podmienky z definície ekvivalencie význačných bodov. Na obrázku 1.4b) je význačný bod p_1 ekvivalentný s p_2 podľa druhej podmienky. Na obrázku 1.4c) je príklad rozmnoženia význačného bodu p_1 na význačné body p_2, p_3 a p_4.

Tvrdenie 1.2.1. Ekvivalencia posunutia význačných bodov nie je tranzitívna.

Dôkaz. Vyplýva z obrázku 1.5. Platia ekvivalencie $p_1 \mapsto p_2$ a $p_2 \mapsto p_3$ ale ekvivalencia $p_1 \mapsto p_3$ neplatí. □



Obr. 1.5: Ukážka, kedy ekvivalencia posunutia význačných bodov nie je tranzitívna.

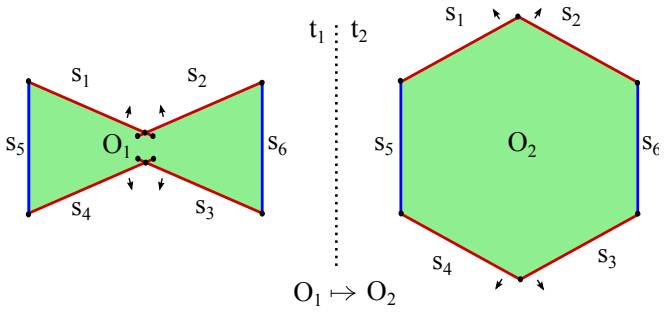
Definícia 1.2.10 (**ekvivalencia posunutia sektorov** $\mapsto$)**.** Hovoríme, že sektory $((x_1, t_1), (y_1, t_1))$ a $((x_2, t_2), (y_2, t_2))$ sú ekvivalentné v rámci ekvivalencie posunutia a zapisovať $((x_1, t_1), (y_1, t_1)) \mapsto ((x_2, t_2), (y_2, t_2))$, ak platí:

$$((x_1, t_1) \mapsto (x_2, t_2) \wedge (y_1, t_1) \mapsto (y_2, t_2)).$$

Pre ekvivalenciu posunutia jednoduchých oblastí je potrebné definovať pre jednoduchú oblasť $O = (V, t)$ *osekanú jednoduchú oblasť* $\tilde{O} = (\tilde{V}, t)$. Vytvárame ju tak, že do zoznamu $\tilde{V}$ priradíme všetky prvky zoznamu V a pokiaľ zoznam význačných bodov $\tilde{V}$ obsahuje za sebou idúcu postupnosť troch význačných bodov, ktoré ležia na jednom senzore, stredný z týchto bodov odstránime z $\tilde{V}$. To robíme, pokiaľ taká trojica existuje.

Týmto spôsobom sú odstránené význačné body, ktoré sú pre definovanie $S(O)$ zbytočné (po ich odstránení sa množina $S(O)$ nezmení). Ak by sme

ich neodstránili potom by nasledovná definícia nedostatočne opisovala, ako intuitívne rozumieme zhode jednoduchých oblastí v rôznych časoch. Problém spočíva v tom, že ak by sme ku každému bodu hľadali ekvivalentný, nie vždy by sa to podarilo. Ak napríklad zoznamu V patria význačné body p_1, p_2 a p_3 (v tomto poradí), ktoré ležia na jednom senzore a p_2 je miesto rozídenia nastávajúceho v čase t, k tomuto bodu už neexistovať význačný bod v čase $t^{'} > t$ v rámci komponentu grafu $G_{S,t'}$ s oblasťou O. (p_2 je artikulácia v grafe $G_{S,t}$.)

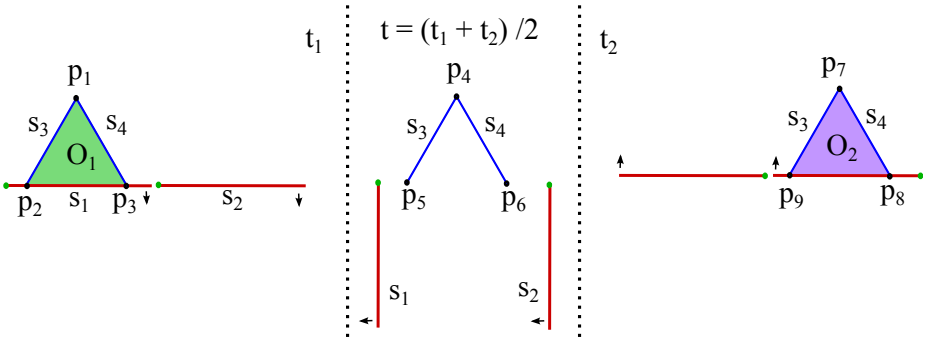


Obr. 1.6: Ekvivalencia posunutia jednoduchých oblastí O_1 a O_2

Definícia 1.2.11 (ekvivalencia posunutia jednoduchých oblastí $\mapsto$). Nech k jednoduchej oblasti $O_1 = (V_1, t_1)$ (resp. $O_2 = (V_2, t_2)$) je osekanou jednoduchou oblasťou $\tilde{O_1} = (\tilde{V_1}, t_1)$ (resp.$\tilde{O_2} = (\tilde{V_2}, t_2)$). Hovoríme, že jednoduché oblasti O_1 a O_2 sú ekvivalentné v rámci ekvivalencie posunutia a zapisovať $O_1 \mapsto O_2$, ak platí:

$$\begin{gathered}\forall t; t_2 < t^{'} < t_1 \; \exists O = (V, t) : \\ \forall p_1 \in \tilde{V_1} \; \exists p_1^{'} \in V : p_1 \mapsto p_1^{'} \wedge \\ \forall p_2 \in \tilde{V_2} \; \exists p_2^{'} \in V : p_2 \mapsto p_2^{'} \wedge \\ \forall p_3 \in V \; \exists p_3^{'} \in V_1 : p_3 \mapsto p_3^{'} \wedge \\ \forall p_4 \in V \; \exists p_4^{'} \in V_2 : p_4 \mapsto p_4^{'}.\end{gathered}$$

Poznámka 1.2.2. Pri implementácii ekvivalencie posunutia jednoduchých oblastí sa zaoberáme prípadom, ak t_1 je predchádzajúci krok k t_2 a t_2 je nasledujúci krok k t_1. V tomto prípade nie je potrebné aby podmienky z definície boli splnené pre $\forall t^{'}; t_2 < t^{'} < t_1$, ale postačí ak $\exists t^{'}; t_2 < t^{'} < t_1$.
Stačí si uvedomiť, že medzi časmi t_1 a t_2 nenastáva žiadna udalosť a teda pre $\forall t^{'}; t_2 < t^{'} < t_1$ je graf $G_{S,t'}$ rovnaký. Ak existuje jednoduchá oblasť v čase medzi t_1 a t_2, ktorá spĺňa podmienky z definície ekvivalencie posunutia jednoduchých oblastí, potom aj v každom čase medzi t_1 a t_2 existuje jednoduchá oblasť spĺňajúca tieto podmienky.



Obr. 1.7: Ukážka, prečo v definícii ekvivalencie posunutia jednoduchých oblastí sú potrebné „medzi-oblasti“.

Z obrázku 1.7 vyplýva, že pre ekvivalenciu dvoch jednoduchých oblastí $O_1 = (V_1, t_1)$ a $O_2 = (V_2, t_2)$ nestačí aby platila podmienka:

$$\forall p_1 \in \tilde{V_1}\ \exists p_1^{'} \in V_2 : p_1 \mapsto p_1^{'} \wedge\ \forall p_2 \in \tilde{V_2}\ \exists p_2^{'} \in V_1 : p_2 \mapsto p_2^{'}.$$

Napriek splneniu tejto podmienky je zjavné, že jednoduché oblasti O_1 a O_2 na obrázku 1.7 nie sú ekvivalentné.

Neexistuje jednoduchá syntaktická definícia ekvivalencie posunutia oblastí. Namiesto toho poskytneme sémantickú, s tým že aktuálny syntaktický test je popísaný v podkapitole 2.5.

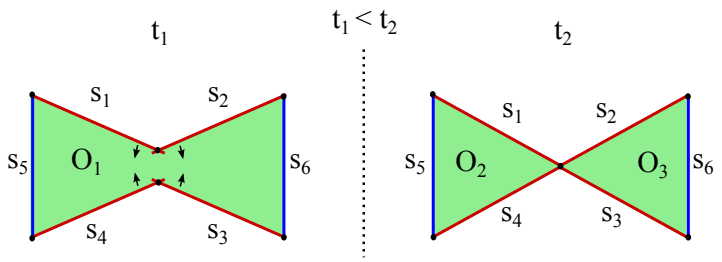
Definícia 1.2.12 (**ekvivalencia posunutia oblastí** $\mapsto$)**.** Hovoríme, že oblasti $O_1 = (Out_1, In_1, t_1)$ a $O_2 = (Out_2, In_2, t_2)$ sú ekvivalentné v rámci ekvivalencie posunutia a zapisujeme $O_1 \mapsto O_2$, ak pre každú funkciu f_I takú, že nenastalo odhalenie votrelca do času $max(t_1, t_2)$ platí:

$$(f_I(t_1) \in S(O_1)) \Leftrightarrow (f_I(t_2) \in S(O_2)).$$

Nasleduje definícia silnej ekvivalencie posunutia $\tilde{\mapsto}$, ktorá je implementovateľná priamočiaro, ale je len špeciálnym prípadom ekvivalencie posunutia oblastí $\mapsto$.

Definícia 1.2.13 (**silná ekvivalencia posunutia oblastí** $\tilde{\mapsto}$)**.** Hovoríme, že oblasti $O_1 = (Out_1, In_1, t_1)$ a $O_2 = (Out_2, In_2, t_2)$ sú ekvivalentné v rámci ekvivalencie posunutia a zapisujeme $O_1 \tilde{\mapsto} O_2$, ak platí, že jednoduchá oblasti (Out_1, t_1) a (Out_2, t_2) sú ekvivalentné a zároveň pre každú jednoduchú oblasť $(F_1, t_1), F_1 \in In_1$ existuje ekvivalentná jednoduchá oblasť $(F_2, t_2), F_2 \in In_2$ a naopak.

Nasledujúcou definíciou formálne popisujeme udalosť rozdelenia oblasti. Pod týmto pojmom si môžeme predstaviť udalosť, pri ktorej sa v jej čase stretnú niektoré senzory a tým jednu oblasť rozdelia na dve, alebo viac menších oblastí. Tieto oblasti vzniknuté rozdelením, pokrývajú množinu bodov, ktorú by pokrývala pôvodná oblasť, ak by nenastalo rozdelenie. Formálne popisujeme túto udalosť pomocou limity.



Obr. 1.8: Rozdelenie oblasti O_1 na oblasti O_2 a O_3.

Definícia 1.2.14 (**Udalosť rozdelenia oblasti**)**.** Udalosť rozdelenia oblasti $O = (Out, In, t^{'})$ nastáva v čase t pokiaľ platí, že v čase t existujú $O_1, ..., O_k (k \geq 2)$ s nenulovými obsahmi a pre $\forall t^{''}; t^{'} < t^{''} < t$ v čase $t^{''}$ existuje oblasť $O^{'}$, že $O \mapsto O^{'}$ a $\lim_{t^{''} \to t^{-}} S(O^{'}) = \bigcup_{i=1}^{k} S(O_i)$.

Oblasti $O_1, ..., O_n$ a k nim ekvivalentné označujeme za oblasti, ktoré vznikli rozdelením oblasti O.

Oblasť môže vzniknúť aj spojením viacerých oblastí. Predstavme si, že oblasti O_1 a O_2 s časom t majú spoločnú hranicu a nastane len jedna udalosť rozídenia senzorov na tejto hranici. Potom oblasti O_1 a O_2 sa spoja do jednej oblasti, ale až v čase $t + \varepsilon$ (pre ľubovoľné $\varepsilon < t_N - t$).

Napriek tomu, že v čase t sa ešte oblasti nespoja do jednej, je tejto udalosti priradený čas t. Udalosť spojenia oblastí aj keď má rovnaký čas ako niektorá udalosť rozdelenia udalostí, v skutočnosti nikdy nenastane v rovnakom čase, ale nastane neskôr „hneď po“ niektorom z časov udalostí rozdelenia oblastí (až vtedy nemajú senzory tvoriace hranicu spoločný bod). V definícii rovnako ako pri udalosti rozdelenia oblasti využívame limitu.

Definícia 1.2.15 (**Udalosť spojenia oblastí**)**.** Udalosť spojenia oblastí $O_1, ..., O_k; k \geq 2$ (s časom t) nastáva v čase t ak existuje oblasť O s časom $t^{'}$, že pre $\forall t^{''} t < t^{''} < t^{'}$ v čase $t^{''}$ existuje oblasť $O^{'}$,že $O \mapsto O^{'}$ a zároveň platí $\lim_{t^{''} \to t^{+}} S(O^{'}) = \bigcup_{i=1}^{k} S(O_i)$.

Oblasť O a k nej ekvivaletné oblasti označujeme za oblasti, ktoré vznikli spojením oblastí $O_1, ..., O_n$.

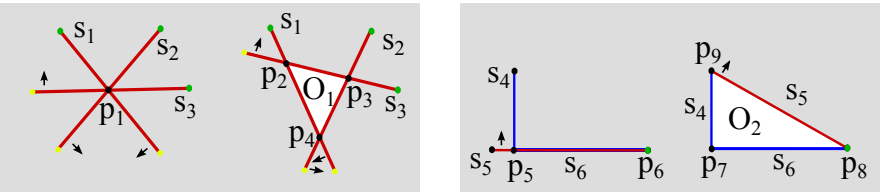
1.2.3 Čistá oblasť

Čistá oblasť pre čas t_0 je taká oblasť $O = (Out, In, t)$, pre ktorú platí, že ak sa votrelec v čase t_0 už nachádza v monitorovanej oblasti, potom sa do oblasti O nemohol dostať bez spozorovania, čiže neexistuje taká trajektória popisujúca jeho pohyb, ktorá by nepretla žiadny zo senzorov.

Definícia 1.2.16 (**Čistá oblasť pre čas** t_0)**.** Oblasť $O = (Out, In, t)$ je čistá pre čas t_0 ak platí:

$$\nexists f_I : f_I(t_0) \neq \perp \wedge f_I(t) \in O \wedge \forall s \in S\ \forall t^{'}; t_0 < t^{'} \leq t : f_I(t^{'}) \notin D_s(t^{'}).$$

Definícia 1.2.17 (**nová čistá oblasť NČO**)**.** Hovoríme, že oblasť $O = (Out, In, t)$ je nová čistá, ak v predchádzajúcom kroku t_B k času t existuje oblasť $O^{'} = (Out^{'}, In^{'}, t_B)$, kde $In^{'} = \emptyset$ a jednoduchý mnohouholník s vrcholmi $V^{'}$ pokrýva nulovú plochu ($S(O^{'}) = \emptyset$) a zároveň platí $O \mapsto O^{'}$.



Obr. 1.9: Nové čisté oblasti O_1 a O_2.

Definícia 1.2.18 (**čas vzniku Oblasti**)**.** Čas vzniku oblasti O je t_O určený nasledovne:

- Ak oblasť je NČO, potom t_O je predchádzajúci krok
- Ak k oblasti O existuje ekvivalentná oblasť $O^{'}$ v predchádzajúcom kroku potom t_0 priradíme čas, ktorý je čas vzniku oblasti $O^{'}$
- Ak oblasť O vznikla rozdelením oblasti $O^{'}$ potom t_0 priradíme čas, ktorý je čas vzniku oblasti $O^{'}$, z ktorej vznikla oblasť O rozpojením
- Ak oblasť O vznikla spojením oblastí, potom t_0 priradíme čas, ktorý je minimum z časov vzniku oblastí, z ktorých vznikla oblasť O spojením.
- Ak oblasť O vznikla spojením oblastí, potom t_0 priradíme čas, ktorý je minimum z časov vzniku oblastí, z ktorých vznikla oblasť O spojením.

Tvrdenie 1.2.2. Oblasť O (pokrývajúca neprázdnu množinu bodov) s časom t je čistá pre čas t_0 rovný času vzniku O vtedy a len v tedy ak spĺňa jednu z nasledujúcich podmienok:

a) O je NČO

b) v predchádzajúcom kroku existuje k O ekvivalentná ČO

c) O vznikla rozpojením čistej oblasti

d) O vznikla spojením len čistých oblastí.

Dôkaz.
$\boxed{\Leftarrow}$

a)

Nech je oblasť O NČO. Podľa definície NČO 1.2.17 existuje v čase t_0 oblasť s nulovým obsahom ekvivalentná oblasti O. Podľa definície ekvivalencie posunutia jednoduchej oblasti 1.2.11 pre každý čas medzi t_0 a t existuje jednoduchá oblasť O' s ekvivalentnými význačnými bodmi a teda v každom čase medzi t_0 a t je oblasť uzavretá. V čase t_0 sa v nej votrelec nachádzať nemohol, keďže mala nulový obsah a vojsť do nej bez spozorovania tiež nemohol, keďže bola stále uzavretá.

b)

Nech je oblasť O ekvivalentná s čistou oblasťou O_c. Potom podľa definícii ekvivalencie posunutia oblastí 1.2.12 je zrejmé, že ak sa votrelec nemôže nachádzať v oblasti O_c, s ktorou je oblasť O ekvivalentná, potom ani v oblasti O sa votrelec nenachádza a teda aj oblasť O je čistá.

c)

Nech oblasť O vznikla rozpojením čistej oblasti O_c. Votrelec sa v čistej oblasti O_c nachádzať nemohol a pokiaľ sa rozpojí na viacero oblastí je zrejmé, že to platí aj pre ne.

d)

Nech oblasť O vznikla spojením len čistých oblastí $O_1, ..., O_n$. Votrelec sa v žiadnej z oblastí $O_1, ..., O_n$ nemôže nachádzať a pokiaľ sa spoja do jednej veľkej oblasti je zrejmé, že ani v tejto oblasti sa nemôže votrelec nachádzať.

$\boxed{\Rightarrow}$

Dokážeme, že ak oblasť vznikla iným spôsobom ako jedným z a), ... , d), potom nemôže byť čistá. Oblasť O s časom t vzniká tromi spôsobmi:

- ekvivalencia s oblasťou s menším časom
- spojením oblastí
- rozdelením oblasti

Rozoberieme prípady, ktoré nespĺňajú žiadnu z možností a), ... , d).

Nech oblasť O s časom t je ekvivalentná s oblasťou O_b s časom t_b ($t_b < t$). Nech oblasť O_b nie je čistá a má nenulový obsah. Potom podľa definície čistej oblasti 1.2.2 existuje funkcia f_I, že v čase t_b sa nespozorovaný votrelec nachádza v oblasti O_b a $f_I(t_0) \neq \perp$. Ak by bola oblasť O čistá, potom sa každý votrelec nespozorovaný do času t v oblasti O určite nenachádza, ak v čase t_0 už bol v monitorovanej oblasti. To je spor s definíciou ekvivalencie posunutia oblastí 1.2.12. Z toho vyplýva, že oblasť O nie je čistá.

Nech vznikla oblasť O rozpojením oblasti O_r, ktorá nebola čistá. Potom sa v nej môže nachádzať votrelec a podľa definície oblasti 1.2.8 sa v nej môže pohybovať bez spozorovania. Takže po rozdelení sa môže nachádzať v ľubovolnej oblasti, ktorá vznikla rozdelním oblasti O_r, takže žiadna oblasť, ktorá vznikla rozdelením nečistej oblasti nie je čistá.

Nech vznikla oblasť O spojením oblastí $O_1, ..., O_n$ a aspoň jedna oblasť O_i nebola čistá. Potom ani O nie je čistá, pretože existuje votrelec, taký, že sa nachádza v oblasti O_i a teda po spojení oblastí sa votrelec môže nachádzať v oblasti O.

□

Kapitola 2

Implementácia

V tejto kapitole sa zaoberáme návrhom algoritmu pre overovanie správnosti siete senzorov, jeho implementáciou a vizualizáciou jeho priebehu, ako aj vizualizácii siete senzorov.

2.1 Popis algoritmu

Na to aby sme určili, či je sieť senzorov navrhnutá správne potrebujeme vedieť pre každý čas povedať, ktoré oblasti v tom čase sú čisté a ktoré sú špinavé. Ak nájdeme čas, o ktorom platí, že všetky oblasti (v monitorovenej oblasti) sú v tomto čase čisté, môžeme povedať, že sieť senzorov je navrhnutá správne a teda po objavení votrelca je s konštantným oneskorením určite spozorovaný.

Aby sme toto vedeli povedať, potrebujeme vedieť identifikovať, ktorá oblasť je *nová čistá oblasť*, ktoré oblasti v rámci dvoch rôznych časov sú ekvivalentné a ktorá oblasť vznikla *rozdelením* inej, alebo ktorá oblasť vznikla *spojením* viacerých oblastí.

K tomu je nevyhnutné vedieť určiť všetky udalosti stretnutia, rozídenia a prekrytia, určiť kedy nastávajú, v ktorom mieste nastávajú a priradiť im ich účastníkov. Ak vieme kedy nastávajú udalosti, môžeme sa pohybovať po diskrétnych hodnotách času. Pre tieto hodnoty vypočítame všetky význačné body, indukovaný graf pre sieť senzorov a tým aj oblasti.

Tieto diskrétne hodnoty času opísané definíciami *predchádzajúci* a *nasledujúci krok* budeme vo všeobecnosti nazývať **kroky**.

Pre každý krok vytvárame *konfiguráciu*. Tá obsahuje všetky potrebné informácie o sieti senzorov, ktorá je natočená o uhol zhodný s časom daného kroku. Konfigurácia s krokom t obsahuje *význačné body,oblasti* a *indukovaný graf* $G_{S,t}$.

Určíme oblasti, ktoré pokrývajú nulovú plochu. Tie potrebujeme k určeniu *nových čistých oblastí*.

Následne môžeme prejsť k najdôležitejšej časti a to je určovanie ekvivalencie oblastí, ich rozdelenie a spájanie medzi jednotlivými krokmi. K tomu však potrebujeme „medzi-kroky“,„pred-kroky“ a „po-kroky“ s konfiguráciami obohatenými o špeciálne indukované grafy. Potreba „medzi-krokov“ vyplýva z poznámky 1.2.2. Potreba „pred-krokov“, „po-krokov“ a špeciálnych indukovaných grafov je vysvetlená v podkapitole 2.4.1.

Všetky tieto kroky možno usporiadať do nasledovného algoritmu OVER ČISTENIE:

Algoritmus OVER ČISTENIE

1. vypočítaj všetky udalosti
   a) vypočítaj udalosti prekrytia senzorov
   b) vypočítaj udalosti pretnutia a rozídenia senzorov
2. vytvor konfigurácie a prídavné konfigurácie k nájdeným krokom
3. pomocou prídavných konfigurácií pre „medzi-kroky“ urči ekvivalencie oblastí
4. pomocou prídavných konfigurácií pre „pred-kroky“ a „po-kroky“ urči rozdelovanie a spájanie oblastí
5. opakuj zadaný počet otočení siete senzorov a kontroluj, ktoré oblasti sú čisté
   - skontroluj, či všetky oblasti sú čisté :
     - ak áno vyhlás, že sieť je správne navrhnutá a skonči
     - ak nie pokračuj
6. vyhlás, že za požadovaný počet otočení nebola monitorovaná oblasť vyčistená sieťou senzorov

Pri kontrolovaní čistoty oblastí sa opierame o tvrdenie 1.2.2 (oblasť je čistá vtedy a len vtedy ...) a pri určovaní ekvivalencii využívame upravenie definície ekvivalencie posunutia jednoduchých oblastí v prípade spomínanom v poznámke 1.2.2.

V nasledujúcich podkapitolách sa zameriame jednotlivo na kroky algoritmu. Spomenieme aj niektoré implementačné problémy, ktoré sa vyskytli a bolo nutné ich riešiť.

2.2 Výpočet udalostí

Ako bolo spomínané, je nevyhnutné identifikovať všetky udalosti prekrytia, pretnutia a rozídenia. Každá z týchto udalostí zahŕňa tri základné informácie:

- uhol udalosti (uhol otočenia siete senzorov, v ktorom nastáva udalosť)
- miesto udalosti
- aktéri udalosti

Miesto udalosti pre udalosti pretnutia a rozídenia je určené jedným bodom. Miesto udalosti pre prekrytie je určené úsečkou.

2.2.1 Zohľadnenie nepresností vo výpočtoch

Pri implementácii je nutné zohľadniť nepresnosť výpočtov, ktoré sú vykonávané na počítačoch. Preto musíme rátať s povolenou odchýlkou, ktorá určuje, že hodnoty vzdialené menej, ako o túto odchýlku, považujeme za rovnaké. Pri implementácii pracujeme s viacerými odchýlkami.

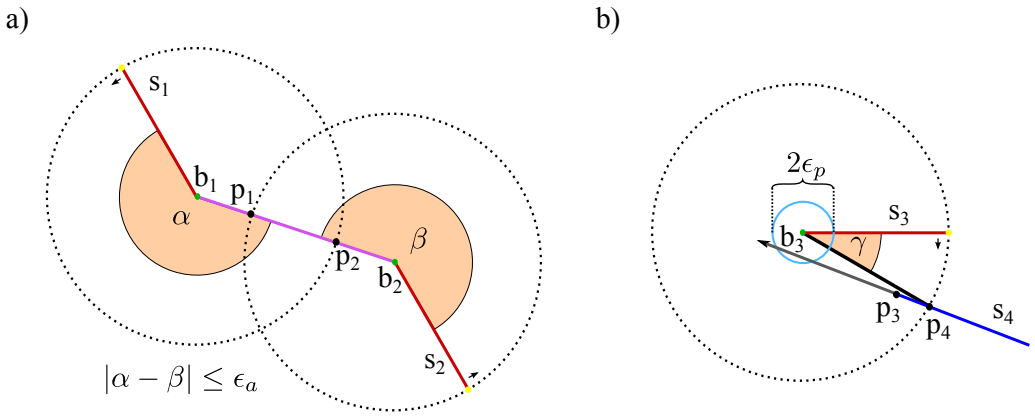
Základnou odchýlkou, s ktorou pracujeme je odchýlka pre bod v euklidovskej rovine a označujeme ju ϵ_p. Ak sú dva body vzdialené od seba o menej ako je hodnota ϵ_p, potom ich považujeme za rovnaké. Podobne pracujeme aj s hodnotami určujúcimi uhol, pre ktoré je zavedené odchýlka ϵ_a.

Tieto odchýlky nepoužívame len na porovnanie dvoch bodov alebo uhlov, ale aj pri iných výpočtoch. Jedným z nich je výpočet udalostí prekrytia senzorov opísaný v nasledovnej časti.

2.2.2 Výpočet udalostí prekrytia

Pri výpočte udalostí prekrytia je potrebné rozobrať zvlášť prípady udalosti prekrytia:

- pre lúčový senzor a rotačný senzor
- pre dva rotačné senzory



Obr. 2.1: Znázornenie výpočtu udalostí prekrytia dvoch rotačných senzorov a rotačného senzoru s lúčovým senzorom.

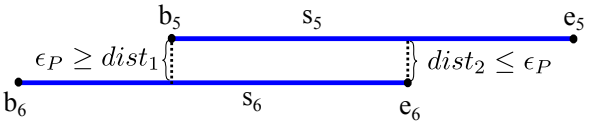
Na výpočte prekrytí možno demonštrovať, ako sa zohľadňuje nepresnosť výpočtov pomocou odchýlok (ϵ_p a ϵ_a). Ak počítame udalosť prekrytia dvoch rotačných senzorov $s_1 = (b_1, e_1, d_1)$ a $s_2 = (b_2, e_2, d_2)$, stačí zistiť či pre vzdialenosť ich začiatočných bodov platí, že $dist(b_1, b_2) < 2r - \epsilon_p$ a zároveň, ak α a β sú také uhly, že α (resp. β) je uhol, o ktorý sa musí s_1 (resp. s_2) otočiť, aby bod b_2 (resp. b_1) ležal na úsečke senzoru s_1 (resp. s_2). Potom musí platiť $|\alpha - \beta| \leq \epsilon_a$. Táto situácia je znázornená na obrázku 2.1 a), kde body p_1 a p_2 sú hraničné body prekrytia, teda úsečka $\overline{p_1, p_2}$ je miestom udalosti, α je uhol udalosti a senzory s_1 a s_2 sú aktérmi udalosti.

Nesmieme však zabudnúť na špeciálny prípad, keď pre vzdialenosť bodov b_1 a b_2 platí, že $dist(b_1, b_2) < r - \epsilon_p$. V tomto prípade môžu nastať až dve udalosti prekrytia. Ak platí $|\alpha - (\beta + 180)| \leq \epsilon_a$, nastávajú dve udalosti prekrytia (s uhlami α a $(\alpha + 180)$).

Pri výpočte udalosti prekrytia rotačného senzora $s_3 = (b_3, e_3, d)$ s lúčovým senzorom $s_4 = (b_4, e_4)$ potrebujeme zistiť, či vzdialenosť úsečky $\overline{b_3, e_3}$ od bodu b_3 je menšia ako $(r - \epsilon_p)$ a zároveň vzdialenosť priamky určenej bodmi b_4 a

e_4 od bodu b_3 je menšia rovná ϵ_p. Táto situácia je znázornená na obrázku 2.1 b), kde body p_3 a p_4 sú hraničné body prekrytia, uhol γ je uhol udalosti a s_3 a s_4 sú aktérmi udalosti.

Ak sa jedná o dva lúčové senzory, nemožno hovoriť o udalostiach, pretože ak sa pretínajú, alebo prekrývajú je to pretrvávajúci stav, ale kvôli prehľadnosti uvádzame výpočet prekrytia lúčových senzorov vrámci výpočtu udalostí prekrytia. Dva lúčové senzory $s_5 = (b_5, e_5)$ a $s_6 = (b_6, e_6)$ sú pokladané za prekrývajúce sa ak vzdialenosť aspoň dvoch bodov z ich hraničných bodov od druhého senzora je menšia ako odchýlka ϵ_P (viď obrázok 2.2). Táto skutočnosť, že senzory s_5 a s_6 sa prekrývajú je zaradená medzi udalosti prekrytia s uhlom otočenia -1, čo znamená, že ide permanentnú udalosť.



Obr. 2.2: Prekrytie dvoch lúčových senzorov

2.2.3 Výpočet udalostí pretnutia a rozídenia

Pri výpočte udalostí pretnutia a rozídenia sme nezvolili priamy spôsob výpočtu, ako pri výpočte udalostí prekrytia. Teda neberieme dvojice, či trojice senzorov, pre ktoré vypočítame, kedy pre ne nastanú udalosti pretnutia, či rozídenia. Implementovanie tohoto spôsobu by bolo náročné, pretože pri tomto postupe je potrebné vyhodnocovať rovnice štvrtého a vyššieho stupňa. Zvolili sme iný spôsob hľadania týchto udalostí, ktorého časová zložitosť je síce väčšia, ale pre našu prácu je vhodnejší.

Pripomeňme si (v skratke), čo hovorí definícia udalosti pretnutia a rozídenia (definície 1.2.1 a 1.2.2). Udalosť pretnutia nastáva, ak senzory nemali spoločný bod a následne majú a udalosť rozídenia v opačnom prípade.

Výpočet udalostí pretnutia a rozídenia počítme aproximančne otáčaním siete senzorov. Pri tomto spôsobe vždy pozeráme na dva uhly otočenia siete senzorov. Ak v pri otočení o menší uhol mali dva senzory priesečník a pri otočení o väčší uhol nemajú priesečník, potom sme našli udalosť rozídenia. Podobne postupujeme pri hľadaní udalosti pretnutia. Ak sme takto na úseku medzi dvomi uhlami našli udalosť, môžeme rozdeliť tento úsek na polovice a pozerať v ktorej z nich nastáva udalosť s presnejším odhadom uhlu udalosti.

Tento postup bol implementovaný s voliteľnou veľkosťou úsekov, po ktorých prechádzame celých 360 stupňov a voliteľným počtom delení úseku na polovice v prípade nájdenia udalosti.

Nemožno zabudnúť na priesečníky troch senzorov. Tie sa vždy nedajú nájsť týmto spôsobom (iba ak nastávajú práve na koncoch kontrolovaných úsekoch). My však tieto udalosti pretnutia troch senzorov potrebujeme pretože určujú oblasti pokrývajúce nulovú plochu, ktoré môžu byť ekvivalentné s oblasťami s nenulovou plochou (*nové čisté oblasti*). Pri vyššie spomínanom hľadaní udalostí pretnutia a rozídenia dvojíc senzorov hľadáme aj trojice senzorov, ktoré majú vzájomné priesečníky a zároveň platí, že tieto priesečníky splynú do jedného bodu skôr, ako je koniec úseku, ktorý práve analyzujeme. Ak sa nachádzame v poslednom delení úseku, tieto tri body označíme za trojicu, s ktorou, ak je niektorá oblasť ekvivalentná, môžeme o nej povedať, že je *novou čistou oblasťou*.

Na určovanie nových čistých oblastí je postačujúce nájsť udalostí prekrytia a udalosti pretnutia troch senzorov. Preto hľadať udalosti pretnutia štyroch a viac senzorov nie je potrebné.

2.3 Vytvorenie konfigurácií

Po tom ako sme našli udalosti prekrytia, pretnutia a rozídenia vytvárame usporiadaný zoznam *krokov*, ktorý pozostáva z uhlov všetkých udalostí utriedených od najmenšieho po najväčší. Následne k týmto *krokom* vytvárame konfigurácie.

Každá konfigurácia obsahuje nasledovné:

- uhol konfigurácie (krok, ku ktorému sme konfiguráciu vytvorili)
- množina význačných bodov konfigurácie (s rovnakým uhlom, ako má konfigurácia)
- indukovaný graf konfigurácie (indukovaný graf pre sieť senzorov s otočením o uhol konfigurácie)
- zoznam oblastí a zoznam prázdnych oblastí konfigurácie (pokrývajúce nulovú plochu)

Pre výpočet význačných bodov je potrebné najskôr určiť, aké informácie má pri implementácii zahŕňať. Význačný bod $p = (x, t)$ má tieto atribúty:

- lokalita (bod x)
- uhol (zhodný s časom t)
- množina vlastníkov: senzory $s_1, ..., s_k$, pre ktoré platí, že $x \in D_{s_i}(t))$
- množina vlastníkov „konca“ : senzory pre ktoré platí, že x je ich koncový bod v čase t
- množina vlastníkov „začiatku“ : senzory pre ktoré platí, že x je ich začiatočný bod
- index (index vrchola, ktorý ho v indukovanom grafe reprezentuje)

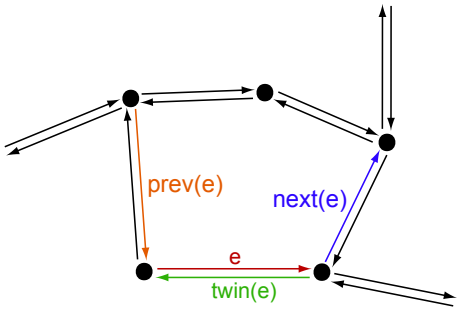
Posledné dve množiny sú dôležité pre hľadanie ekvivalencie význačných bodov podľa prvej podmienky definície 1.2.9 (konce, alebo začiatky rovnakého senzoru v rôznych časoch).

Množinu význačných bodov pre konfiguráciu sa vytvárame nasledovne: otočíme sieť senzorov o uhol konfigurácie, vypočítame všetky priesečníky a prekrytia. Každý senzor má zoznam priesečníkov a hraničných bodov prekrytí, ktoré na ňom ležia. Zoberieme senzor zo siete senzorov a body na ňom ležiace pridávame do množiny význačných bodov konfigurácie s tým, že im priradíme vlastníkov a ak je tento bod koncom (resp. začiatkom) senzora pridáme tento senzor medzi vlastníkov „konca“ (resp. „začiatku“).

Podobne vytvárame indukovaný graf. Body, ktoré ležia na senzoroch stačí vrámci každého senzoru usporiadať podľa vzdialenosti od začiatočného bodu. Tým sú nájdené sektory a z nich je vytvorenie indukovaného grafu priamočiare. Každý vrchol má svoj jedinečný index, ktorý je následne priradený význačnému bodu, ktorý tento vrchol v grafe reprezentuje.

2.3.1 Hľadanie jednoduchých oblastí konfigurácie

Pre hľadanie stien grafu je indukovaný graf prispôsobený tomu, aby sme s ním mohli pracovať ako s omedzenou (bez stien priradeným polhranám) HDS-štruktúrou („Halfedge Data Structures“, ktorá je známa aj ako FE-štruktúra [5]). V tejto štruktúre je každej polhrane e priradená polhrana nasledovná ($next(e)$), predchádzajúca ($prev(e)$) a opačná ($twin(e)$). Ukážka tejto štruktúry je na obrázku 2.3.



Obr. 2.3: HDS-štruktúra použitá na hľadanie stien grafu.

Takéto prispôsobenie grafu je vhodné na hľadanie stien v grafe. Pri hľadaní stien postupujeme nasledovne:

1. ak sme sme po polhrane prešli, označíme ju za navštívenú
2. kým nie sú navštívené všetky polhrany:
   a) zvolíme nenavštívenú hranu a pokračujeme po nasledovnej hrane až kým neprídeme k už navštívenej hrane (pričom si pamätáme vrcholy po ktorých sme prešli)
   b) ak sme prišli k už navštívenej hrane prejdená cesta je nová nájdená stena, ktorú si uložíme (ako zoznam vrcholov)

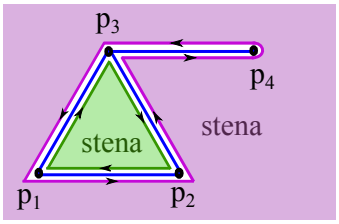
Výsledkom takého prechádzania grafu sú všetky vonkajšie a vnútorné steny indukovaného grafu konfigurácie.

Pre jednoduchú oblasť $O = (V, t)$ je potrebné mať jej tri interpretácie:

- pôvodná interpretácia: zoznam význačných bodov V
- interpretácia osekanej jednoduchej oblasti $\tilde{O} = (\tilde{V}, t)$ k oblasti O: zoznam význačných bodov $\tilde{V}$
- interpretácia ako jednoduchý mnohouholník, ktorý pokrýva $S(O)$: zoznam význačných bodov $\hat{V}$.

Vytvorenie osekanej oblasti $\tilde{O} = (\tilde{V}, t)$ sme už spomínali a tiež aj jej využitie pri hľadaní ekvivalencií posunutia jednoduchých oblastí (1.2.9).

Ostáva nám popísať spôsob vytvorenia tretej interpretácie a to zoznamu význačných bodov $\hat{V}$. Ten vytvoríme podobne ako zoznam $\tilde{V}$, ale odstránime všetky vrcholy, ktoré nezmenia množinu $S(O)$. Táto interpretácia je používaná



Obr. 2.4: Dve steny indukovaného grafu pre sieť so štyrmi lúčovými senzormi.

pri vizualizácii jednoduchých oblastí a určovaní, či sa jedná o vonkajšiu, alebo vnútornú jednoduchú oblasť.

Pri určovaní, či sa jedná o vnútornú alebo vonkajšiu oblasť postupujeme nasledovne: pre každý komponent indukovaného grafu konfigurácie nájdeme jednoduchú oblasť O, pre ktorú platí, že každý význačný bod $p_i = (x_i, t)$ (vrchol) tohoto komponentu platí, že buď $p_i \in V$, alebo $x_i \in S(O)$. Oblasť O spĺňajúca túto podmienku je vonkajšia a ostatné patriace komponentu sú vnútorné. Vzťah $x_i \in S(O)$ vieme určovať algoritmom pre určenie polohy bodu x_i vzhľadom na jednoduchý mnohouholník zadaný jeho vrcholmi (zoznamom $\hat{V}$), pri ktorom počítame počet pretnutí jeho hranice (podrobnejšie v publikácii [4]).

2.3.2 Vytvorenie (zložitejších) oblastí

Nech konfigurácia má jednoduché oblasti $O_1, ..., O_n$, potom oblasti konfigurácie vytvárame nasledovným postupom:

1. pre každú jednoduchú oblasť konfigurácie $O_i = (V_i, t)$ vytvoríme oblasť $O_i^{'}$, takú že $O_i^{'} = (Out_i, In_i, t)$ a zoznam Out_i je zoznamom V_i.

2. pre každú oblasť $O_i^{'} = (Out_i, In_i, t)$ prejdeme každú jednoduchú vonkajšiu oblasť $O_{outer} = (V_{outer}, t)$ a ak platí $\forall (x_i, t) \in V_{outer} : x_i \in S(O_i^{'})$, potom V_{outer} pridáme do množiny In_i

Takto sa do množiny In_i pridajú aj také jednoduché vonkajšie oblasti $O_1^x, ..., O_k^x$, pre ktoré platí, že $\exists O^{''} \in In : S(O_i^x) \subset S(O^{''})$. Na obrázku 2.5 je pre oblasť $O_3^{'}$ takouto jednoduchou vonkajšou oblasťou (Out_1, t). Tieto jednoduché vonkajšie oblasti odstránime, aby bola splnená definícia oblasti:

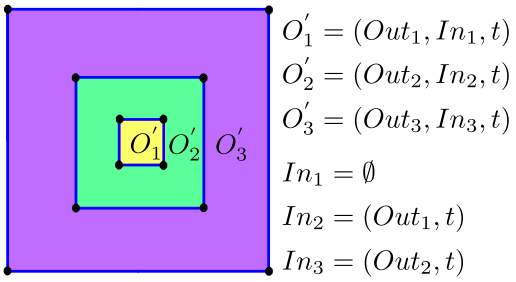
1. ku každej oblasti $O_i^{'}$ vytvoríme prázdnu množinu $Del_{O_i^{'}}$

2. prejdeme ešte raz každú oblasť $O_i^{'} = (Out_i, In_i, t)$

   - pre každý zoznam $V_{in} \in In_i$ nájdeme oblasti (ktoré sú „vo vnútri“ niektorých iných oblastí zachytených v In_i) $O_1^x = (Out_{O_1^x}, In_{O_1^x}, t)$, ..., $O_m^x = (Out_{O_m^x}, In_{O_m^x}, t)$, ktoré patria rovnakému komponentu grafu ako jednoduchá vonkajšia oblasť (V_{in}, t). Potom do množiny $Del_{O'}$ pridáme prvky zjednotenia $In_{O_1^x} \cup ... \cup In_{O_m^x}$.

3. každej oblasť $O_i^{'} = (Out_i, In_i, t)$ z množiny In_i odstránime prvky, ktoré obsahuje jej množina $Del_{O_i'}$

Týmto postupom sú zostrojené všetky oblasti konfigurácie .



Obr. 2.5: Nájdené oblasti $O_1^{'}$, $O_2^{'}$ a $O_3^{'}$ s viacnásobným vnorením.

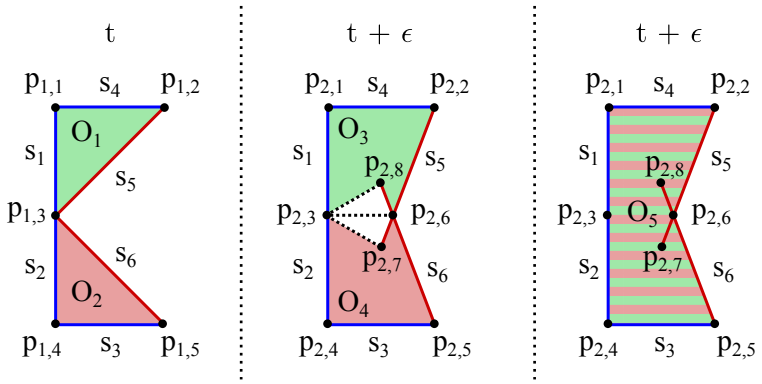
2.4 Vzťahy jednoduchých oblastí

V tejto časti popisujeme, ako identifikovať pre dve nasledovné konfigurácie, v akom vzťahu sú ich jednoduché oblasti. Môžu byť ekvivalentné, viaceré mohli vzniknúť rozdelením jednej jednoduchej oblasti z predchádzajúcej konfigurácie, alebo jedna jednoduchá oblasť mohla vzniknúť spojením viacerých jednoduchých oblastí z predchádzajúcej konfigurácie. Problém je, že niektoré oblasti mohli vzniknúť kombináciou spojenia oblastí a rozdelenia (obrázok 2.8).

Pre zistenie ekvivalencie pre dve za sebou idúce konfigurácie s uhlami α_1 a α_2 určujeme „medzi-krok“ s uhlom v strede medzi α_1 a α_2 a vytvárame k nemu prídavnú konfiguráciu. Ďalej je implementácia ekvivalencie jednoduchých oblastí priamočiara vďaka jej definícii a definícii osekanej jednoduchej oblasti. Podrobnejšie rozoberieme identifikáciu rozdelenia a spájania jednoduchých oblastí v nasledovných podkapitolách.

2.4.1 Identifikácia spojenia jednoduchých oblastí

Pri spájaní jednoduchých oblastí skúmame ako sa jej hranica môže „pretrhnúť“, na ktorých miestach, a s ktorými oblasťami sa spojí do jednej.

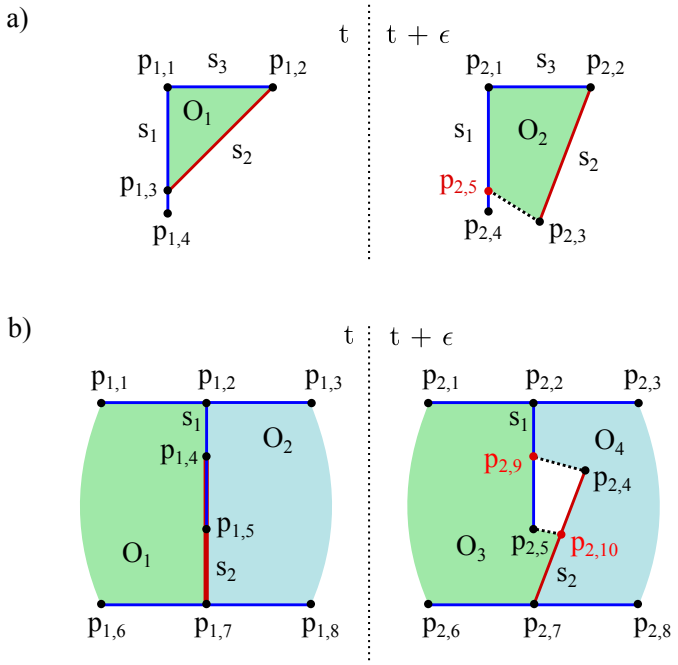


Obr. 2.6: Znázornenie identifikácie spájania O_1 a O_2 do oblasti O_5, pomocou špeciálnych jednoduchých oblastí O_3 a O_4. Keďže pre znázornené jednoduché oblasti platí $(O_1 \mapsto O_3) \wedge (O_2 \mapsto O_4) \wedge (S(O_3) \subset S(O_5)) \wedge (S(O_4) \subset S(O_5))$ z toho vieme, že O_1 a O_2 sa spájajú do oblasti O_5.

Nech čas udalosti spojenia je t a dve za sebou nasledujúce konfigurácie, ktoré sledujeme, majú uhly α_1 (prislúchajúci času t) a α_2. Podľa definície udalosti spojenia sa potrebujeme pozerať na čas medzi α_1 a α_2, ale taký, ktorý je čo najbližšie k času t, teda vytvoríme „po-krok“ $(\alpha_1 + \epsilon)$ a k nemu prídavnú konfiguráciu K_a. Podstatné je si uvedomiť, čo sa pri pretrhnutí hranice deje. Jeden význačný bod sa rozdelí na dva (alebo viac), ktoré už nemajú spoločnú hranu. My tieto hrany simulujeme (špeciálnymi hranami) a tým získame dôležitú informáciu o tom, ktoré oblasti z prechádzajúcej konfigurácie sa spájajú. Na obrázku 2.6 je vidieť pridané špeciálne hrany medzi vrcholy reprezentujúce význačné body $p_{2,3}$, $p_{2,6}$, $p_{2,7}$ a $p_{2,8}$. Kompletný graf, ktorý medzi týmito vrcholmi vznikol znázorňuje spoločného predka $p_{1,3}$ z predchádzajúcej konfigurácie.

Nesmieme zabudnúť aj na udalosti rozídenia senzorov, ktorých miestom udalosti nie sú hraničné body jeho aktérov (senzorov). Nech senzory s_1 a s_2 majú priesečník v čase udalosti ich rozídenia, ktorý je koncový bod s_2, ale nie je hraničným bodom s_1. Vtedy v prídavnej konfigurácii pridávame špeciálny význačný bod na senzor s_1 so vzdialenosťou od začiatočného bodu, ako je od neho vzdialené miesto udalosti rozídenia. Táto situácia je znázornená na obrázku 2.7 a). Nakoniec pridaný špeciálny význačný umelo označíme za ek-

vivalentný s význačným bodom miesta udalosti, aby pri vytváraní špeciálneho grafu (popísaného nižšie) bola pridaná špeciálna hrana.



Obr. 2.7: Pridávanie špeciálnych význačných bodov ($p_{2,5}$, $p_{2,9}$ a $p_{2,10}$) a špeciálnych hrán ($(p_{2,5}, p_{2,3})$, $(p_{2,9}, p_{2,4})$, $(p_{2,5}, p_{2,10})$).

Podobne, ako pre udalosť rozídenia, pridávame špeciálne význačné body aj pri udalostiach prekrytia. Ak sme pridali dva vrcholy (ani jeden hraničný bod miesta prekrytia sa nezhodoval s hraničnými bodmi oboch senzorov) ako je znázornená na obrázku 2.7 b), je ťažšie určiť, ktoré jednoduché oblasti znázorňujú nulovú plochu, ako napríklad jednoduchá oblasť ohraničená bodmi $p_{2,9}$, $p_{2,5}$, $p_{2,4}$ a $p_{2,10}$. Takéto štvorice si pamätáme a ak sa jednoduchá oblasť nachádza medzi nimi znázorňuje miesto prekrytia (nie je pokladaná za jednoduchú oblasť je odstránená).

Špeciálny indukovaný graf pre prídavnú konfiguráciu k „po-kroku“ vytvárame nasledovne:

1. skopírujeme hrany a vrcholy pôvodného indukovaného grafu tejto prídavnej konfigurácie

2. pridáme vrcholy, reprezentujúce pridané špeciálne význačné body

3. nech dva vrcholy v_1 a v_2 reprezentujúce význačné body p_1 a p_2, ktoré majú spoločného predka p_s (význačný bod z predchádzajúcej konfigurácie ekvivalentný s p_1 a p_2) s indexom g , potom vrcholy v_1 a v_2 spojíme špeciálnou hranou s indexom g.

V takto vytvorenom špeciálnom grafe medzi vrcholmi so spoločným predkom vytvoríme kompletný graf. Tento kompletný graf reprezentuje jeden význačný bod, a teda každá jednoduchá vnútorná oblasť je odstránená, pokiaľ jej všetky vrcholy majú spoločného predka a aspoň jedna z hrán na jej hranici je špeciálna s indexom tohoto predka. Takto ostanú len jednoduché oblasti, ktoré voláme špeciálne jednoduché oblasti(na obrázku 2.6 sú to jednoduché oblasti O_3 a O_4).

Po odstránení oblastí, ktoré znázorňujú jeden bod, alebo miesto prekrytia, nám ostali len špeciálne jednoduché oblasti. K nim nájdeme ekvivalentné jednoduché oblasti z predchádzajúcej konfigurácie. Potom sa pozrieme na pôvodný indukovaný graf prídavnej konfigurácie a pôvodné jednoduché oblasti (na obrázku 2.6 je to oblasť O_5). Potom, pre pôvodné jednoduché oblasti O_1^{orig}, ..., O_n^{orig} a špeciálne oblasti O_1^{spec}, ..., O_m^{spec} vieme povedať, z ktorých špeciálnych oblastí je tvorená pôvodná oblasť. Ak platí, že $S(O_i^{spec}) \subset S(O_j^{orig})$ potom pridáme jednoduchú oblasť O_i^{spec} k jednoduchým oblastiam z ktorých je jednoduchá oblasť O_j^{orig} tvorená.

Takto pre každú pôvodnú jednoduchú oblasť O_j^{orig} je určená množina špeciálnych jednoduchých oblastí, z ktorých je tvorená. Na základe ekvivalencií posunutia medzi špeciálnymi jednoduchými oblasťami a jednoduchými oblasťami predchádzajúcej konfigurácie určíme aj množinu jednoduchých oblastí z predchádzajúcej konfigurácie, z ktorých jednoduchá oblasť O_j^{orig} vzniká spojením.

Problém, ktorý sa pri vytváraní špeciálneho grafu objavuje, je pretínanie hrán špeciálnymi hranami, čím sa ruší planarita špeciálneho indukovaného grafu a hľadanie stien a špeciálnych jednoduchých oblastí by nefungovalo. Preto musíme tento problém odstrániť. Nájdeme špeciálne hrany pretínajúce hranu, ktorá nie je špeciálna. Na miesto priesečníku týchto hrán pridáme vrchol. Špeciálnu hranu rozdelíme na dve špeciálne s rovnakými indexmi a hranu, ktorá nebola špeciálna rozdelíme na dve, ktoré nie sú špeciálne.

2.4.2 Identifikácia rozdelení jednoduchých oblastí

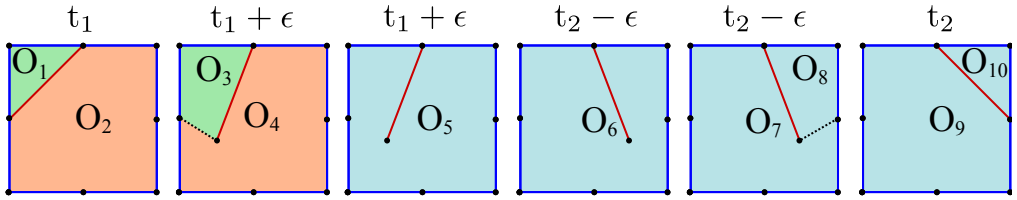
Pre identifikáciu rozdelenie oblastí je voči identifikácii spájania oblastí rozdiel v tom, že postupujeme opačným smerom. Tak ako sme pre dve za se-

bou idúce konfigurácie s uhlami α_1 a α_2 vytvárali „po-krok“ , tak vytvoríme „pred-krok“ $(\alpha_1 - \epsilon)$ a k nemu prídavnú konfiguráciu. A vytvoríme podobným spôsobom špeciálny graf, ale špeciálne vrcholy nie sú pridávané na základe udalostí rozídenia a prekrytia, ale na základe udalostí pretnutia a prekrytia a špeciálne hrany nie sú pridávané, ak je nájdený spoločný predok, ale spoločný nasledovník.

Potom, ako sme naznačili, pokračujeme opačným smerom. Pre špeciálne jednoduché oblasti zistíme, z ktorých pôvodných oblastí vzniknú rozdelením a určíme ktoré jednoduché oblasti z nasledujúcej konfigurácie sú ekvivalentné so špeciálnymi oblasťami z prídavnej konfigurácie pre „pred-krok“.Na základe toho určujeme pre každú jednoduchú oblasť nasledujúcej konfigurácie pôvodnú oblasť konfigurácie pre „pred-krok“ z ktorej vznikla rozdelením.

2.4.3 Určenie vzťahov

Ostáva nám dať do jedného celku postup identifikácie spojení a rozdelení jednoduchých oblastí. Zoberme dve po sebe idúce konfigurácie K_1 a K_2. Vytvoríme k tejto dvojici prídavné konfigurácie K_a (k uhlu „po-krok“), K_m (k uhlu medzi uhlami konfigurácii K_1 a K_2) a K_b (k uhlu „pred-krok“). Keďže medzi konfiguráciami K_1 a K_2 nenastáva žiadna udalosť, z toho je zjavné, že indukované grafy pre všetky konfigurácie vytvorené pre uhly medzi uhlami konfigurácií K_1 a K_2 sú zhodné. Takto vieme jednoducho pre prídavné konfigurácie určiť ekvivalencie ich jednoduchých oblastí (sú ekvivalentné ak majú zhodné zoznamy V).

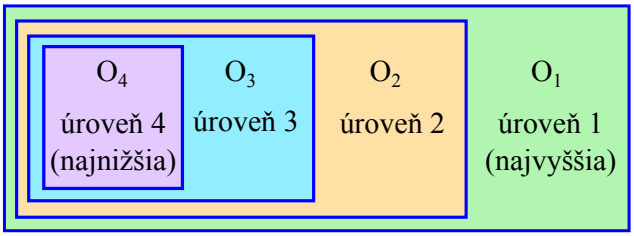


Obr. 2.8: Vzťahy: O_5 je spojením O_1 a O_2, O_9 a O_10 vznikli rozdelením O_6.

Takže ak určíme, ku konfigurácii K_a z ktorých jednoduchých oblastí konfigurácie K_1 vznikli jej jednoduché oblasti, potom sa táto informácia ekvivalenciou medzi jednoduchými oblasťami konfigurácií K_a a K_b prenesie na jednoduché oblasti konfigurácie K_b. Následne sa určí z ktorých jednoduchých oblastí konfigurácie K_b vznikajú jednoduché oblasti konfigurácie K_2.

2.5 Vzťahy oblastí

Pre každú oblasť určujeme jej úroveň vnorenia (viď obrázok 2.9). Nasleduje neformálne definovanie usporiadanie týchto úrovní. Oblasť s najvyššou úrovňou je taká, ktorá nie je obsiahnutá žiadnou inou. Oblasti určené jej množinou In sú o jednu úroveň nižšie. Takto pokračujeme až k najnižšej úrovni, ktorá obsahuje oblasti, ktorých množina In je prázdna.



Obr. 2.9: Usporiadanie úrovní vnárania oblastí.

Pri analýze oblastí, ich čistoty a zisťovaní vzťahov oblastí medzi za sebou idúcimi konfiguráciami, postupujeme podľa potreby buď od úrovne najnižšej až k úrovni najvyššej, alebo opačným smerom.

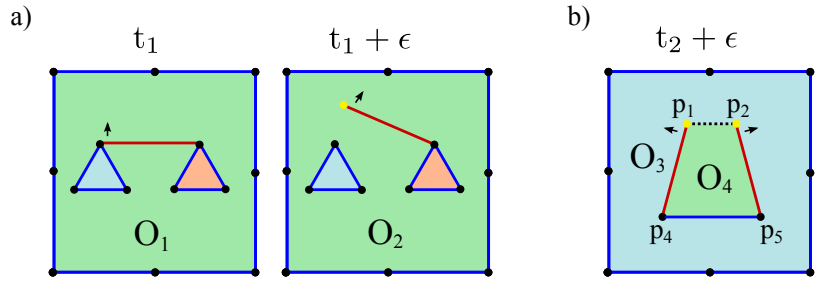
2.5.1 Identifikácia spojenia oblastí

Pri identifikácii spájania oblastí postupujeme od najnižšej úrovni po najvyššiu, keďže vďaka množinám In vieme s oblasťami pracovať ako s dátovou štruktúrou strom. Ak ideme po vetve, striedajú sa uzly dvoch typov. Jedným typom je uzol znázorňujúci komponent indukovaného grafu a druhý znázorňujúci oblasť. Algoritmom do hĺbky sa vieme dostať k najnižším úrovniam a začať ich riešiť ako prvé.

Tak, ako sa vytvárajú z jednoduchých oblastí (zložitejšie) oblasti, rovnakým spôsobom sú vytvárané špeciálne oblasti zo špeciálnych jednoduchých oblastí. Majme konfiguráciu K_a vytvorenú pre „po-krok“ k uhlu konfigurácie K. Nájdeme špeciálne oblasti a pôvodné oblasti.

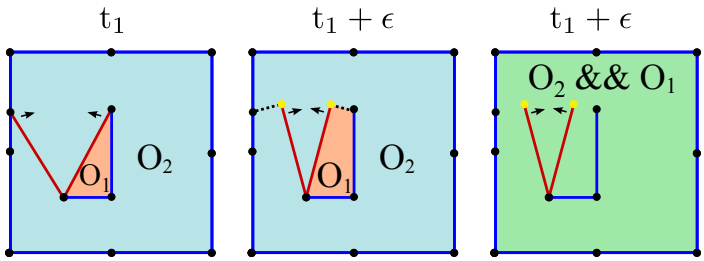
Najprv určíme spájanie pôvodných oblastí v rámci jedného komponentu (neberieme do úvahy ich In množiny) na základe spájania ich jednoduchých oblastí Out. Potom skontrolujeme, či sa nepripoja (pod)oblasti z In množiny skontrolovaním, či existuje taká špeciálna hrana, ktorá by prepájala oblasť s jej (pod)oblasťou z In množiny. Takáto hrana je na obrázku 2.10 b) medzi

význačnými bodmi p_1 a p_2. Ak existuje tak oblasti, ktoré prepája sa spájajú do jednej (na obrázku sa spájajú oblasti O_3 a O_4).



Obr. 2.10: Na obrázku a) je znázornený príklad, kedy platí, že $O_1 \mapsto O_2$ a zároveň neplatí $O_1 \not\mapsto O_2$. Na obrázku b) je znázornená špeciálna hrana medzi p_1 a p_2, ktorá určuje že oblasti O_3 a O_4 sa spájajú.

Ostáva nám spomenúť prípad, ak sa z komponentu, ktorý práve prezeráme odtrhne časť a zaradí sa do množiny In niektorej pôvodnej oblasti. Tento prípad nastáva vtedy, keď sa zväčšila mohutnosť množiny In pôvodnej oblasti v porovnaní s mohutnosťou množiny In jej príslušnej špeciálnej oblasti (viď obrázok 2.11). Zoberieme všetky prvky množiny In (aj nové), pridáme len medzi ich vrcholy špeciálne hrany zo špeciálneho indukovaného grafu a potom opäť zisťujeme, či sa spájajú so špeciálnou oblasťou do jednej pôvodnej, na základe existencie spomínaných špeciálnych prepájacích hrán (2.10 b)).



Obr. 2.11: Roztrhnutie komponentu grafu a pridanie do množiny In a súčasné spojenie dvoch oblastí do jednej.

2.5.2 Identifikácia rozdelení oblastí

Pri identifikácii rozdelenia oblastí opäť postupujeme obrátene ako pri určovaní spájania oblastí. Postupujeme od najvyššej úrovni až po najnižšiu. A ako pri

spájaní oblastí nás zaujímal prípad, keď sa odtrhla časť komponentu a pridala sa do množiny In niektorej oblasti komponentu, pri rozdeľovaní oblastí nás zaujíma opačný jav. Keď sa zmenší mohutnosť množiny In niektorej špeciálnej oblasti voči k nej prislúchajúcej pôvodnej oblasti vďaka spojeniu vnútorného komponentu s vonkajšou jednoduchou oblasťou. Stačí si opäť uvedomiť symetriu so spájaním oblastí na základe ktorej sa rozdeľovanie oblastí identifikuje.

2.5.3 Identifikácia ekvivalencie oblastí

Silná ekvivalencia posunutia $\tilde{\mapsto}$ oblastí je implementovateľná bez problémov, lenže nepopisuje všetky možnosti, kedy oblasť ostáva „tou istou“ oblasťou. Na obrázku 2.10 a) nie sú podľa definície $\tilde{\mapsto}$ oblasti O_1 a O_2 ekvivalentné, ale podľa definície ekvivalencie posunutia oblastí $\mapsto$, sú tieto oblasti ekvivalentné. Ako sme už spomínali platí vzťah $(O_1 \mapsto O_2) \Rightarrow (O_1 \tilde{\mapsto} O_2)$.

Testovanie ekvivalencie posunutia oblastí $\mapsto$ v rámci prídavnej konfigurácie pre „po-krok“ spočíva, ako pri spájaní a rozpájaní, v kontrole existencie špeciálnych prepájacích hrán spomínaných pri identifikácii spojenia (2.10 b)).

Ekvivalenciu oblasti neurčujeme medzi dvoma konfiguráciami priamo pomocou „medzi-konfigurácie“, ale postupne pri určovaní spájania a rozpájania pomocou prídavných konfigurácií pre „po-krok“ a „pred-krok“.

Pri určovaní ekvivalencie medzi oblasťami konfigurácie K a oblasťami k nej prídavnej konfigurácie pre „po-krok“ K_a postupujeme, ako pri identifikácii spájania oblastí, od najnižšej úrovne po najvyššiu. Na najnižšej úrovne majú pôvodné oblasti prázdnu množiny In. Určíme ekvivalenciou jednoduchých oblastí pre množiny Out špeciálnych oblastí, s ktorými oblasťami z konfigurácie K sú špeciálne oblasti ekvivalentné. Po odstránení špeciálnych hrán si musíme dať pozor na odtrhnutie komponentu, ktorý sa pridá do množinu In, ako sme spomínali pri identifikácii spájania oblastí. Pôvodná oblasť $O = (Out, In, t)$ bude ekvivalentná so špeciálnou $O_s = (Out_s, In_s, t)$, ak množina Out bude taká podmnožina Out_s, že v ich rozdiely budú len tie vrcholy, ktoré patria niektorej množine z In a navyše platí, že žiadna (pod)oblasť určená množinou In sa nemá spomínanú prepájajúcu hranu (2.10 b)).

Pri určovaní ekvivalencie medzi oblasťami konfigurácie K a oblasťami k nej prídavnej konfigurácie pre „pred-krok“ K_a si stačí opäť uvedomiť, že

tento postup je len obráteným postupom ako pri určovaní ekvivalencie medzi konfiguráciou a k nej prídavnej konfigurácie pre „po-krok“. Postupujeme, ako pri identifikácii rozpájania oblastí, od najvyššej úrovni až po najnižšiu a nesledujeme odtrhávanie komponentov a ich pridávanie do množiny In, ale pripájanie komponentov z množiny In ku komponentu, ktorému patrí množina Out. Potom pomocou existencie spomínaných prepájacích hrán určujeme, či sa jedná o ekvivalenciu oblastí.

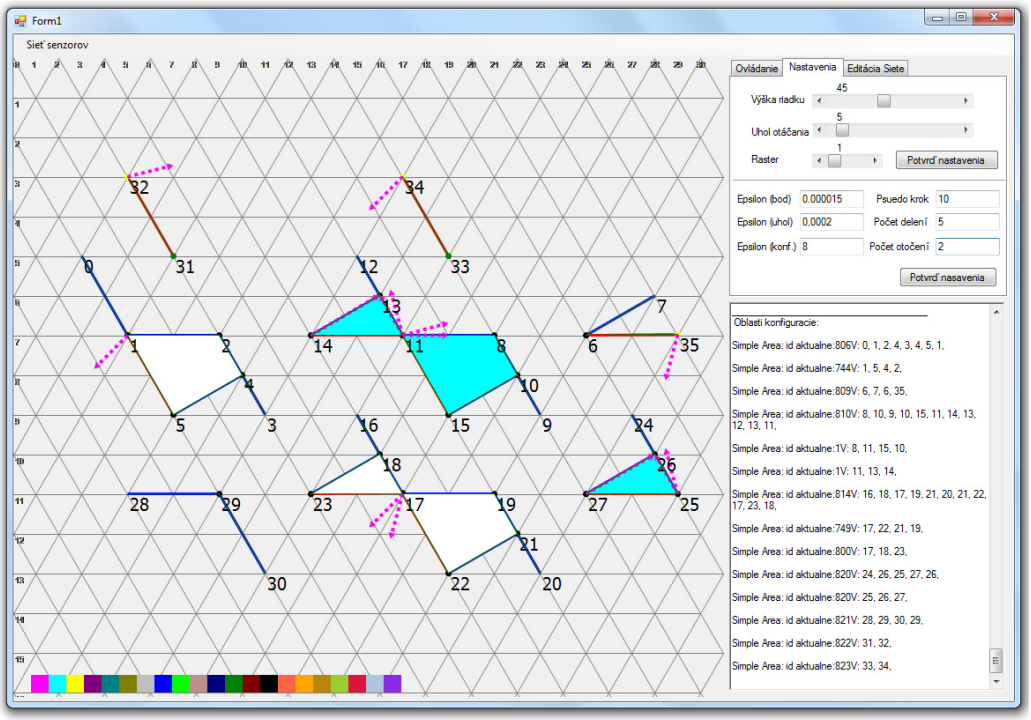
2.6 Čistenie oblastí

Majme po sebe idúce konfigurácie K_1 a K_2. A k tejto dvojici majme prídavné konfigurácie K_a (k uhlu „po-krok“), K_m (k uhlu „medzi-krok“) a K_b (k uhlu „pred-krok“). Ak by pre každú oblasť konfigurácie K_2 platilo, že vznikla buď len spájaním, alebo len rozdelením oblasti, alebo je ekvivalentná s niektorou oblasťou K_1, potom by platil nasledovný vzťah:
(vlastnosť byť čistou oblasťou pre oblasť O značíme $C(O)$)

- ak oblasť O je nová čistá nastav $C(O) = true$
- ak oblasť O je ekvivalentná s oblasťou O_B z predchádzajúcej konfigurácie potom $C(O) = C(O_B)$
- ak oblasť O vznikla z viacerých oblastí $O_1, O_2, ..., O_k$ potom $C(O) = C(O_1) \wedge C(O_2) \wedge ... \wedge C(O_k)$
- ak oblasť O vznikla rozdelením oblasti O_R z predchádzajúcej konfigurácie potom $C(O) = C(O_R)$.

Avšak niektoré oblasti nespĺňajú danú podmienku. Môže sa stať, ze niektoré oblasti vzniknú spojením viacerých oblastí do jednej a jej následným rozdelením, a preto si pomáhame konfiguráciou K_m. Ak sa pozeráme na vzťah medzi oblasťami konfigurácií K_1 a K_m (resp. K_m a K_2), potom je už daná podmienka splnená a môžeme uplatniť spomenuté pravidlá prenášania vlastnosti byť čistou oblasťou.

2.7 Vizualizácia



Obr. 2.12: Ukážka aplikácie pre overovanie a vizualizáciu siete senzorov.

Zvolili sme trojuholníkovú sieť na ktorej možno editovať sieť senzorov (pridávať, alebo mazať senzory). Sieť možno vytvoriť editovaním prázdnej, alebo načítať z XML súboru so správnym formátom (podľa priložených ukážkových xml súborov). Vizualizácia je prispôsobená tomu, aby sme mohli pozorovať detaily siete senzorov (vieme ju zväčšovať a posúvať). Rovnako vieme sledovať aj detaily indukovaného graf a špeciálneho indukovaného grafu. Grafy vieme sledovať samostatne bez zobrazenia senzorov. Rozlíšené sú pôvodné a špeciálne hrany v špeciálnych indukovaných grafoch. Farebne sú vyplnené vnútorné jednoduché oblasti. Pokiaľ ich farba výplne je biela, znamená to, že ide o čisté oblasti. Ak sú farebné, jedná sa o špinavé (nie čisté) oblasti, pričom sú používané rôzne farby, aby bolo možné sledovať vzťahy medzi oblasťami rôznych konfigurácií. V základnom nastavení aplikácie sa vypĺňajú oblasti farebne len v uhloch konfigurácií a prídavných konfigurácií. V uhloch konfigurácií sú prerušovanými šípkami znázornené aj pohyby význačných bodov (ich ekvivalencia s význačnými bodmi nasledovnej konfigurácie).

Aplikácia pre vizualizáciu siete senzorov a overovanie jej návrhu ponúka nie len nastavenie detailu pre pozorovanie, ale aj možnosť prispôsobiť si a nastaviť vlastné hodnoty odchýlok ϵ_p, ϵ_a a veľkosť hodnoty ϵ použitej na vytváranie „po-krokov“ a „pred-krokov“. Nastaviteľný je aj počet otočení siete senzorov a veľkosť úsekov, ktoré sa kontrolujú pri hľadaní udalostí rozídenia a pretnutia a počet iterácií (počet delení kontrolovaných úsekov). Pri editovaní siete senzorov je možné pridávať senzory len na vrcholy trojuholníkov trojuholníkovej siete a preto je nastaviteľná hustota trojuholníkov v trojuholníkovej sieti (raster a výška trojuholníkov).

Záver

V našej práci sme sa venovali navrhnutiu a implementovaniu algoritmu pre overovanie korektnosti návrhu siete senzorov pre monitorovanie roviny a jednoduchých mnohouholníkov použitím rotačných a lúčových senzorov. Venovali sme sa aj vizualizácii siete senzorov a dôležitých momentov algoritmu overovania. Ako prvé sme formálne popísali model, s ktorým sme pracovali. Tento model sme pre účely automatického overovania rozšírili o nové pojmy, ktoré podrobnejšie popisujú konfiguráciu systému.

Definovali sme pojmy, ktoré nám umožňujú sledovať dynamické správanie systému a vyhodnotiť, v ktorých oblastiach sa môže votrelec nachádzať a v ktorých nie. Medzi tieto pojmy patria ekvivalencie posunutia pre význačné body, sektory, jednoduché oblasti a oblasti, udalosti stretnutia a rozídenia senzorov a udalosti spojenia a rozdelenia oblastí.

Následne sme vyslovili tvrdenie o vlastnosti byť čistou oblasťou. Toto tvrdenie sme využili na zisťovanie, ktoré oblasti sú v rámci konfigurácii čisté. V implementačnej časti sme sa venovali problémom, ktoré neboli priamočiaro implementovateľné a vyžadovali hlbšie rozobratie. Navrhli sme algoritmické riešenie overovania správnosti návrhu siete senzorov, pričom niektoré kroky algoritmu sú riešené aproximačnými metódami. Zohľadnené boli aj nepresnosti výpočtov v počítači. V implementácii pre nedostatok času nie je zahrnutá práca so vzťahmi medzi (zložitejšími) oblasťami. Podarilo sa nám implementovať algoritmus overovania pre jednoduché oblasti, resp. oblasti s prázdnou množinou In, a vďaka tomu bolo ukázané, že hlavné myšlienky algoritmu overovania sú implementovateľné a fungujú v našej aplikácii správne.

Pre množstvo problémov, ktoré sa vyskytli pri návrhu a implementácii algoritmu overovania siete senzorov, a ktorým sa bolo nutné dlhodobejšie venovať, nezostal priestor na riešenie ďalších cieľov zo zadania tejto diplomovej práce.

Literatúra

[1] Stefan Dobrev, Lata Narayanan, a Jaroslav Opatrny. *Optimal Sensor Networks for Monitoring a Plane using Rotating and Beam Sensors* . Institute of Mathematics, Slovak Academy of Sciences, Bratislava, Slovakia. Department of CSE, Concordia University, Montreal, QC, Canada, 2012.

[2] Pavel Chalmovianský, Andrej Ferko, Roman Galbavý , Ľudovít Niepel.*Zložitosť geometrických algoritmov* : skriptá. Bratislava: FMFI UK, 2001.

[3] Michiel Smid *Computing intersections in a set of line segments: the Bentley-Ottmann algorithm*, 2003

[4] Michael Galetzka, Patrick O. Glauner *A correct even-odd algorithm for the point-in-polygon (PIP) problem for complex polygons*, 2012

[5] K. Weiler. *Edge-based data structures for solid modeling in a curved surface environment.* IEEE Comput. Graph. Appl., 5(1):21-40, 1985.

[6] Yann Chevaleyre, Francois Sempe, and Geber Ramalho. A Theoretical Analysis of Multi-Agent Patrolling Strategies. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3* (AAMAS '04), Vol. 3. IEEE Computer Society, Washington, DC, USA, 1524-1525, 2004.

[7] Nicola Basilico, Nicola Gatti, and Francesco Amigoni. Developing a Deterministic Patrolling Strategy for Security Agents. *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 02* (WI-IAT '09), Vol. 2. IEEE Computer Society, Washington, DC, USA, 565-572, 2009.

[8] Joseph O'Rourke. *Art Gallery Theorems and Algorithms.* Oxford University Press, Inc., New York, NY, USA, 1987.

[9] Prosenjit Bose, Thomas Shermer, Godfried Toussaint and Binhai Zhu. *Guarding Polyhedral Terrains* , 1992.

[10] Ulrike Bartuschka, Kurt Mehlhorn a Stefan Naher. A robust and efficient implementation of a sweep line algorithm for the straight line segment intersection problem. *Proc. workshop on algor. engineering, Venice, Italy*, strany 124-135, 1997.

[11] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar. Reliable density estimates for coverage and connectivity in thin strips of finite length.*Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, strany 75–86, 2007.

[12] B. Bhattacharya, M. Burmester, Y. Hu, E. Kranakis, Q. Shi, and A. Wiese. Opti-mal Movement of Mobile Sensors for Barrier Coverage of a Planar Region. *In pro-ceedings of 2nd Annual International Conference on Combinatorial Optimization and Applications (COCOA'08) held August 21-24, 2008, in St. John's, Newfound-land, Canada. Lecture Notes in Computer Science*, 2008.

[13] A. Chen, S. Kumar, and T.H. Lai. Designing localized algorithms for barrier coverage.*Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, strany 63–74, 2007.

[14] J. Czyzowicz, E. Kranakis, D. Krizanc, I. Lambadaris, L. Narayanan, J. Opatrny, L. Stacho, J. Urrutia, and M. Yazdani. On minimizing the maximum sensor move-ment for barrier coverage of a line segment. In ADHOC-NOW, strany 194–212, 2009.

[15] J. Czyzowicz, E. Kranakis, D. Krizanc, I. Lambadaris, L. Narayanan, J. Opatrny, L. Stacho, J. Urrutia, and M. Yazdani. On minimizing the sum of sensor movements for barrier coverage of a line segment. In Ioanis Nikolaidis and Kui Wu, editors, *Ad-Hoc Now, volume 6288 of Lecture Notes in Computer Science*, strany 29–42. Springer Verlag, 2010.

[16] C. F. Huang and Y. C. Tseng. The coverage problem in a wireless sensor network. In WSNA*'03: Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, pages 115–121, New York, NY, USA, 2003. ACM.

[17] R. Kershner. The Number of Circles Covering a Set. *Amer. J. Math.*, 61:665–671, 1939.

[18] S. Kumar, T.H. Lai, and A. Arora. Barrier coverage with wireless sensors. *Wireless Networks*, 13(6):817–834, 2007.

[19] X. Li, H. Frey, N. Santoro, and I. Stojmenovic. Localized sensor self-deployment with coverage guarantee. *ACM SIGMOBILE Mobile Computing and Communica-tions Review*, 12(2):50–52, 2008.

[20] K. Scheibe, F. Huang, and R. Klette. Pose estimation of rotating sensors in the context of accurate 3d scene modeling. *Journal of Universal Computer Science*, 16(10):1269–1290, 2010.