* 1. **Matematická morfologie**

Jelikož tato práce využívá některé koncepty matematické morfologie, bylo by vhodné popsat, o co se vlastně jedná.

Matematická morfologie je samostatnou oblastí, která se zabývá analýzou obrazu. Využívá algebru nelineárních operací, které jsou aplikovány na tvar objektu v obraze. Tato oblast se dá považovat za předchůdce lineárního algebraického systému: lineární kombinace, známé též jako konvoluce. Využití je poměrně široké a dá se najít například v metodách redukce šumu, segmentace objektů z popředí, nebo kvantitativním popisu objektů [2].

Předpoklad, pro využití morfologických operací, je možnost popsat obraz pomocí bodových množin jakékoli dimenze. Například to může být n-dimenzionální euklidovský prostor. Přirozenou doménou pro popsání obrazu je euklidovský dvourozměrný prostor **E2**. Morfologické operace aplikujeme v digitálním protějšku euklidovského prostoru **Z2** - Tj., pracujeme s celými čísly [2].

Zásadním pojmem je strukturní element (Obr. 1).

A picture containing shape

Description automatically generated

Obrázek 1 - příklad strukturních elementů

1. **Algoritmus pro vyhledávání nejužších míst ve 2D**
   1. Otázka propustnosti

Důležitá otázka zní, jak definujeme propustný materiál. Pro tuto práci si určíme, že propustný materiál je takový, který umožňuje průchod z jednoho konce materiálu na druhý přes vnitřní póry a praskliny. Toto ujasnění propustnosti je důležité, jelikož do jisté míry určuje, jakým způsobem vytvoříme následující algoritmy pro vyhledání nejužších míst v materiálu. V našem případě budeme počítat s tím, že vstupem pro algoritmus budou pouze obrazová data naskenovaného materiálu. Nebudeme vyžadovat od uživatele vstupní bod (seed) pro vyplňování. Raději vyhledáme body na okraji materiálu, ze kterých jsme schopni určit, zda je materiál propustný. Případně další vstup navíc by mohlo být označení, na které stěně krychle materiálu hledáme body počátku propustnosti. S takovýmto vstupem bychom mohli ještě pracovat.

* 1. Zamyšlení nad přístupem k problému.

Způsoby, jak vyhledat nejužší místa v propustném materiálu, můžeme vymyslet více. Nejjednodušší algoritmus, který by se dal použít, by mohl při vyplňování zároveň prohledávat okolní čtyři, nebo osm sousedních pixelů zrovna vyplňovaného pixelového bodu. Narážíme zde ovšem na více problémů. Ten první je, že by nestačilo prosté kontrolování sousedního okolí pixelu, ale bylo by nutné prohledávat větší oblast, jelikož nemůžeme předpokládat, že v materiálu je nejužší místo velikosti 1 pixelu. Další problematická situace nastává při určité konfiguraci hraniční oblasti.

Diagram

Description automatically generated with medium confidence Qr code

Description automatically generated

Obrázek 2: 4 směry hledání Obrázek 3: 8 směrů hledání

Problémové případy:

Chart, waterfall chart

Description automatically generated Chart

Description automatically generated

Obrázek 4: problém 4 směrů hledání Obrázek 5: problém 8 směrů hledání

Velký problém, na který narážíme u jednoduchého prohledávání sousedních pixelů je ten, že existují případy, kdy je nejužší místo těžko detekovatelné. Příklad můžeme vidět na obrázcích výše. Pokud bychom vyhledávali pouze do 4 směru, tak nedetekujeme nejužší místo mezi dvěma obdélníky. Ovšem i když zařídíme, abychom prohledávali ne do čtyř, ale do osmi směrů, negarantujeme tím úspěšnou detekci. V případě druhého obrázku můžeme vidět, že nejužší místo mezi dvěma trojúhelníky tento přístup neodhalí. Tyto případy jsou tedy důvodem, proč využití této jednoduché metody není dobrý nápad. Další způsoby, jak prohledávat okolí jednoho bodu, jistě existují, ovšem komplexnost začíná růst jak z implementačního hlediska, tak výpočetního. Přesuneme se tedy na další možnost řešení daného problému.

Chart, diagram

Description automatically generated

* 1. Základní postup pro řešení problému

1. Nalezení všech unikátních počátečních ploch na stěně materiálu, které propojují jednu stěnu materiálu se stěnou opačnou.
2. Extrakce celého propojujícího prostoru mezi stěnami pomocí algoritmu vyplnění.
3. Aplikace algoritmu na vyhledávání nejužších míst.
4. Souřadnicový výstup se zaznačenými nejužšími místy v materiálu.
   1. Další možné řešení problému

Pokud nejsme reálně schopni využít prohledávání sousedního okolí kolem pixelového bodu, což by byl nejjednodušší přístup, vymyslíme jiný.

Následující přístup využívá teorie matematické morfologie, a to konkrétně dilatace. Myšlenka spočívá v tom, že místo toho, abychom zkoumali jednotlivé pixely, zda se nejedná o nejužší místa, raději postupně dilatujeme hraniční body. Tento přístup je méně výhodný v tom, že bude nutné opakované vyplňování zkoumané oblasti, abychom zjistili, zda došlo k uzavření materiálu. Tato operace rozšiřování bude opakována tak dlouho, až dojde k uzavření. Jakmile zjistíme, že již nelze materiálem prostoupit na jeho druhý konec, víme, že jsme zaplnili místo, které je nejužší. Následně stačí si toto místo vyhledat (například pomocí průniku předchozí a poslední vyplněné oblasti před uzavřením) a označit. Tímto způsobem získáme souřadnici (souřadnice, je-li takovýchto míst více) nejužšího místa v propustném materiálu. Problémem tohoto přístupu je ovšem to, že nalezneme nejužší místa, které nemusejí být zásadní pro celkovou propustnost materiálu. Jinak řečeno, výsledkem by byly i souřadnice úzkých míst, vedoucích do různých “kapes“ v materiálu. My ovšem stojíme pouze o úzká místa, která ovlivňují celkovou propustnost materiálu.

* 1. Přístup řešení propustnosti s pomocí metody black box

Kdybychom vynechali potřebu zjišťovat souřadnice konkrétních úzkých míst v materiálu, mohli bychom se soustředit pouze na krajní oblasti začátek – konec. V tomto přístupu bychom opět využili metody postupné dilatace hraničních pixelů, k určení velikosti nejužšího místa. Tento přístup by byl vhodný v případě, že by nám stačilo znát, jak propustný materiál je, nikoli kde je propustný.

Příklad rozdělení počátečních pixelů na levé straně do unikátních skupin (3 skupiny) a určení jejich protilehlých oblastí, které získáme po vyplnění.

A picture containing chart

Description automatically generated