Bakalářská práce

České vysoké učení technické v Praze

F3

Fakulta elektrotechnická
Katedra kybernetiky

Detekce objektů z hloubkové kamery

Lukáš Kunt

Vedoucí: Dr. Petr Štěpán
Studijní program: Kybernetika a robotika
Květen 2020

Poděkování

Prohlášení

Abstrakt

Klíčová slova: slovo, klíč

Vedoucí: Dr. Petr Štěpán
Ústav X,
Uliční 5,
Praha 99

Abstract

Keywords: word, key

Title translation: Object Detection from Depth camera — Journey to the who-knows-what wondeland

Obsah

Obrázky

Tabulky

Kapitola 1
Matematický aparát

1.1 Homogenní souřadnice

Během celé práce budu pracovat jak s kartézskými, tak i s homogenními souřadnicemi. Ty jsou v Euklidovském prostoru dimenze n reprezentovány pomocí vektoru, který má $n+1$ prvků. Je-li $\boldsymbol{r}$ bod v trojdimenzionálním Euklidovském prostoru $\mathbb{E}^3$ reprezentován parametry x, y, z, pak pro převod mezi kartézskými a homogenními souřadnicemi platí následující vztahy.

$$\mathbf{r} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \rightarrow \mathbf{r}_{hom} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1.1}$$

$$\mathbf{r}_{hom} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \rightarrow \mathbf{r} = \begin{bmatrix} x/w \\ y/w \\ z/w \end{bmatrix} \tag{1.2}$$

Pomocích homogenních souřadnic můžeme sestavit transformační matici $\mathbf{A}$, tato se skládá z matice rotace $\mathbf{R}$ a vektoru translace $\mathbf{t}$.

$$\mathbf{A} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{1.3}$$

1.2 Rotace a rotační matice

Natočení souřadného systému, popřípadě objektu s tímto svázaným vhledem k jinému systému v prostoru dimenze n se nejjednodušeji popíše pomocí rotační matice $\mathbf{R} \in \mathbb{R}^n$. Mezi nejběžnější rotační matice patří rotace o úhel θ kolem os kartézského souřadného systému v $\mathbb{E}^3$. Tyto matice vypadají následovně.

$$\mathbf{R_x} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.4}$$

$$\mathbf{R_y} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.5}$$

$$\mathbf{R_z} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.6}$$

Obecné natočení pak můžeme dostat složením 3 a více těchto rotací [1] Pro vytvoření matice rotace $\mathbf{R}$ v prostoru $\mathbb{R}^3$ kolem obecné osy dané normalizovaným vektorem $\mathbf{u}$ o úhel θ slouží Rodriguesův vzorec [20].

$$\mathbf{R} = \mathbf{I} + \tilde{\mathbf{u}}\sin\theta + \tilde{\mathbf{u}}^2(1-\cos\theta); \quad \tilde{\mathbf{u}} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \tag{1.7}$$

Chceme-li natočit vektor $\mathbf{a}$ tak, aby byl rovnoběžný s vektorem $\mathbf{b}$, použijeme vzorec 1.7, úhel rotace θ a vektor $\tilde{\mathbf{u}}$, kolem kterého rotace proběhne, se určí následovně.

$$\tilde{\mathbf{u}} = \frac{\mathbf{a}\times\mathbf{b}}{\|\mathbf{a}\times\mathbf{b}\|} \tag{1.8}$$

$$\theta = \arccos\left(\frac{\mathbf{a}\cdot\mathbf{b}}{\|\mathbf{a}\|\,\|\mathbf{b}\|}\right) \tag{1.9}$$

Podle [10] je tato metoda nevhodná pro výpočet na počítači. Zejména pak v případech, kdy bychom měli počícat arccos hodnot blízko ± 1. Toto je způsobeno omezenou přesností reprezentace desetinných čísel v počítači (takzvaná floating point aritmetika). Přesnější je tedy použít tento vzorec.

$$\theta = \operatorname{atan2}\left(\|\mathbf{a}\times\mathbf{b}\|, \mathbf{a}\cdot\mathbf{b}\right) \tag{1.10}$$

Výhoda funkce atan2 oproti arcus tangens je ošetřené dělení nulou, ke kterému může dojít při zaokrouhlování desetinných čísel. Navíc funkce vrací úhel v rozmezí $[0, 2\pi]$

1.3 Sobelův operátor pro detekci hran

Hrana v obrazu může být také pozorována jako prudká změna intenzity obrazu v daném místě. Rychlé změny se dají detekovat pomocí gradientu, který je pro spojitou funcki $f(x, y)$ v bodě (x, y) vyjádřen následovně.

$$\nabla f(x,y) = \begin{bmatrix} \mathbf{G}_x & \mathbf{G}_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix} \tag{1.11}$$

[1] Na pořadí zde záleží. Není možné dostat libovolnou rotační matici, pokud budou dvě po sobě doucí rotace probíhat kolem stejné osy

Velikost a směr gradientu se určí následovně.

$$\mathrm{mag}(\nabla f) = ||\nabla f||_2 = \sqrt[2]{\mathbf{G}_x^2 + \mathbf{G}_y^2} \tag{1.12}$$

$$\boldsymbol{\theta} = \arctan(\frac{\mathbf{G}_y}{\mathbf{G}_x}) \tag{1.13}$$

Parciální derivace musí být spočítané pro každý pixel (x, y) obrazu $\mathbf{A}$. K tomuto se využívá Sobelových kernelů $\mathbf{K}_x$, $\mathbf{K}_y$. Tyto se konvolvují s obrazem a v každém bodě aproximují hodnotu obou parciálních derivací $g_x(x, y), g_y(x, y)$, ze kterých se pak podle vztahu 1.11 určí gradient.

$$\mathbf{K}_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & -1 \end{bmatrix} \quad \mathbf{K}_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{1.14}$$

$$g_x(x, y) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} \mathbf{K}_x(k, l)\mathbf{A}(x + k, y + l) \tag{1.15}$$

$$g_y(x, y) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} \mathbf{K}_y(k, l)\mathbf{A}(x + k, y + l) \tag{1.16}$$

$$\tag{1.17}$$

Určení gradientu intenzity tímto způsobem je relativně nenáročné. Pro zrychlení výpočtu se používá aproximace magnitudy gradientu místo exaktního výpočtu 1.12. Nejjednodušší aproximací je

$$\mathrm{mag}(\nabla f) \approx ||\nabla f||_1 = |\mathbf{G}_x| + |\mathbf{G}_y|. \tag{1.18}$$

Ta dává sice velice nepřesný výsledek, ale pro aplikace, kde stačí hrubý odhad velikosti gradientu, může značně snížit výpočetní dobu.

1.4 Hledání konvexního obalu množiny bodů

Je-li dána množina S obsahující n bodů v euklidovském prostoru $\mathbb{E}^3$, popřípadě euklidovské ploše $\mathbb{E}^2$, pak konvexní obal H množiny S je nejmenší konvexní možina obsahující množinu S [7].

Mezi nejpoužívanější algoritmy pro hledání konvexního obalu bodů v euklidovské ploše $\mathbb{E}^2$ patří následující.

1.4.1 Grahamův algoritmus

Je nalezen bod $\mathbf{p}_0$, jehož y-ová souřadnice nabývá nejmenší hodnoty ze všech bodů množiny S. Všechny ostatní body jsou seřazeny podle úhlu, který svírají s osou x. K tomuto se využívá obecného třídícího algoritmu, jako je například heapsort, a tento krok má tedy časovou náročnost $\mathcal{O}(n \log n)$. Seřazené body jsou postupně procházeny a na základě následujícího kritéria zařazeny[2], popřípadě vyloučeny z konvexního obalu H [15]

$$\begin{gathered} O(\mathbf{p},\mathbf{r},\mathbf{q}) = \det\begin{pmatrix} p_x & r_x & q_x \\ p_y & r_y & q_y \\ 1 & 1 & 1 \end{pmatrix} \\ \begin{Bmatrix} \mathbf{r} \in H, \mathbf{p} = \mathbf{r}, \mathbf{r} = \mathbf{q}, \mathbf{q} = \mathbf{p_i} \text{ iff } O > 0 \\ \mathbf{r} \notin H, \mathbf{p} = \mathbf{p}, \mathbf{r} = \mathbf{q}, \mathbf{q} = \mathbf{p_i} \text{ iff } O \leq 0 \end{Bmatrix}, \end{gathered} \tag{1.19}$$

kde $\mathbf{p}_i$ je další bod v pořadí seřazených bodů z S. Výše popsané ověřování bodů je provedeno v čase $\mathcal{O}(n)$.

Na podobném principu funguje i Andrewův algoritmus. Ten využívá lexikografického uspořádání bodů podle x-ové souřadnice, čímž se vyhne výpočtům s destinou čárkou, které mohou být zdrojem nepřesností.[6]

1.4.2 Jarvisův pochod (Jarvis march)

Je zvolen bod $\mathbf{p}_0$ náležící konvexnímu obalu H, což je například první bod v lexikografickém uspořádání bodů S. Ten je najit v čase $\mathcal{O}(n)$. Dále je spočítán relativní úhel mezi $\mathbf{p}_0$ a ostatními body. Z těch je pak vybrán bod $\mathbf{p}_n$, jehož úhel nabývá nejmenší hodnoty. Bod $\mathbf{p}_n$ je přídán do obalu H a postup se opakuje z bodu $\mathbf{p}_n$. Algoritmus končí ve chvíli, kdy platí $\mathbf{p}_n = \mathbf{p}_0$. Tento postup odpovídá postupnému procházení všech vrcholů polygonu, který reprezentuje konvexní obal, a jeho časová náročnost je $\mathcal{O}(hn)$, kde h je počet vrcholů. Může tedy být pro aplikace s předem známým počtem vrcholů rychlejší než Grahamův algoritmus.

1.4.3 Chanův algoritmus

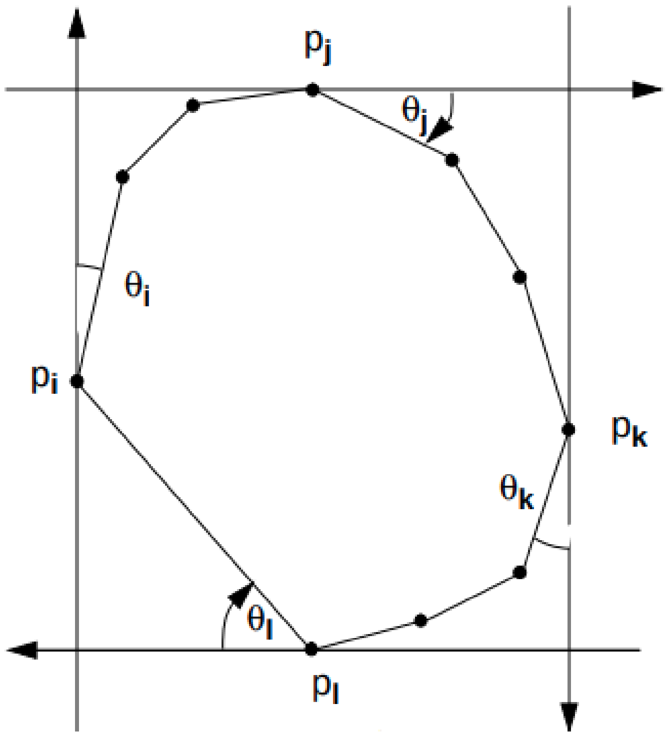
Jedná se o ideální na výstup citlivý algoritmus určený k výpočtu komplexního obalu množiny. Výpočet dosahuje náročnosti $\mathcal{O}(n \log h)$

Množina bodů S je rozdělena do K množin Q_i. V každé množině se nachází maximálně m bodů, kde m je předem určená hodnota, pro kterou musí platit $m \geq h$. Pokud tato rovnost není splněna, musí být hodnota m zvýšena a algoritmus se opakuje. Následně je proveden Grahamův algoritmus na každou množinu Q_i s časovou náročností $\mathcal{O}(n \log m)$. Poté je zvolen bod, který náleží H a je užit Jarvisův pochod. Ten musí najít nejmenší prvek mezi maximálně m seřazenými vrcholy v K množinách, což je realizovatelné v čase $\mathcal{O}(K \log m)$. Pro h vrcholů je tedy výsledná časová náročnost $\mathcal{O}(n \log h)$, za předpokladu, že hodnota m je blízká hodnotě h.[7]

1.5 Nejmenší ohraničující obdélník

Nejmenší ohraničující obdélník je možné určit z konvexního obalu H množiny S. Ten může být najit v čase $\mathcal{O}(h)$ například pomocí star algoritmu [17] nebo pomocí otáčejících třemenů (rotating calipers).

[2] Bod zařezený do konvexního obalu může být v dalším kroku vyloučen.



Obrázek 1.1: Otáčející se třmeny

Kolem množiny jsou umístěny dva páry na sebe ortogonální přímek, takzvaných třmenů (viz. obrázek 1.1), které se konvexního obalu dotýkají v některém z vrcholů $\mathbf{p}_i$, nebo mají společnou přímku spojující vrcholy $\mathbf{p}_{i-1}$ a $\mathbf{p}_i$. V každém kroku je spočítán úhel θ_i mezi třmenem, který se dotýká vrcholu $\mathbf{p}_i$, a vrcholem $\mathbf{p}_{i+1}$. Tento výpočet se opakuje pro všechny 4 třmeny, viz. obrázek 1.1. Následně je výbrán úhel $\theta = \min\{\theta_i, \theta_j, \theta_k, \theta_l\}$, o který je provedena rotace všech třmenů. Třmeny jsou pak ve směru svého normálového vektoru posunty tak, aby platila výše uvedená podmínka doteku třmenu s konvexní množinou v jednom, resp. dvou vrcholech. Dále je vypočten obsah obdélníku, jehož strany tvoří výše zmíněné třmeny. Celý postup se poté opakuje, dokud není vyzkoušeno všech h obdélníků. Následně je vybrán ten s nejmenší plochou.

1.6 Vlastní čísla a vektory

Pro čtvercovou matici $\mathbb{A} \in \mathbb{R}^{n \times n}$ a nenulový vektor $\mathbf{v} \in \mathbb{R}^n$ a sklár $\lambda \in \mathbb{R}$ platí

$$\mathbf{Av} = \lambda \mathbf{v}. \tag{1.20}$$

Skalár λ se nazývá vlastní číslo matice $\mathbf{A}$ a vektor $\mathbf{v}$ vlastní vektor příslušný vlastnímu číslu λ. Spektrální věta (viz. [21]) říka: Je-li matice $\mathbf{A}$ symetrická, pak má n vlastních čísel, kde $n = \mathrm{rank}\mathbf{A}$ a všechna vlastní čísla jsou reálná.

1.7 Singulární rozklad

add some changes

Singulární SVD[3]rozklad umožní každou matici $\mathbf{A} \in \mathbb{R}^{m \times n}$ rozložit jako

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T \tag{1.21}$$

kde matice $\mathbf{S}$ je diagonální, matice $\mathbf{U}$ je ortogonální a obsahje vlastní vektory matice $\mathbf{A}\mathbf{A}^T$, obdobně je pak i matice $\mathbf{V}$ ortogonální a je složena z vlastních vektorů matice $\mathbf{A}^T\mathbf{A}$

1.8 Analýza hlavních komponent

Analýza hlavních komponent (PCA[4])

[3]Z anglického Singular Value Decomposition
[4]Z anglického Principal Component Analysis

Kapitola 2

Kamera

2.1 Dírkový model kamery

Kamera zobrazuje body euklidovského prostoru $\mathbb{E}^3$, které jsou popsány pomocí světových souřadnic na body v euklidovské ploše $\mathbb{E}^2$, která se nazývá obraz. Body obrazu se označují jako pixely.

Nejjednodušším modelem kamery je model dírkový, který má optické centrum, neboli centrum projekce, v konečnu. Centrum projekce umístíme do počátku kartézského souřadného systému a budeme uvažovat plochu kolmou na osu Z našeho souřadného systému, kterou nazveme obrazovou rovinou. Ta je popsána jediným parametrem a to ohniskovou vzdáleností f. V dírkovém modelu kamery se bod v prostoru $\mathbf{p} = (x, y, z)^T$ promítne na bod obrazu pomocí zobrazení

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \mapsto \begin{bmatrix} fx/z \\ fx/z \end{bmatrix}, \tag{2.1}$$

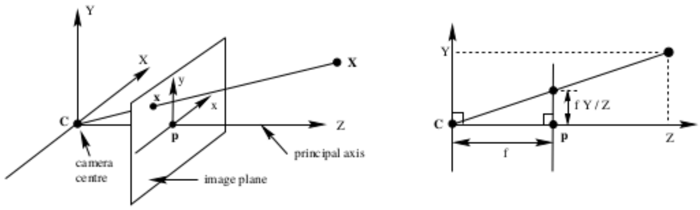
což odpovídá bodu, kde přímka spojující centrum projekce a bod v prostoru, protne obrazovou rovinu (viz obrázek 2.1). Ve vztahu 2.1 jsme předpokládali, že počátek souřadného systému obrazu je v optickém středu. Konvencí však je za počátek souřadnic zvolit levý horní roh obrazu. K zápisu využijeme homogenních souřadnic a dostaneme následující rovnici

pro zobrazení z souřadného systému kamery do systému obrazu s počátkem v levém horním rohu obrazu

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \mapsto \begin{bmatrix} fx + zc_x \\ fy + zc_y \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{2.2}$$

kde jsme jako c_x a c_y označili souřadnice optického středu. Označíme-li obecný bod v prostoru, který je popsán v souřadném systému kamer, $\mathbf{x}_p$ a odpovídající bod v rovině obrazu $\mathbf{x}$, pak můžeme rovnici 2.2 přepsat jako

$$\mathbf{x} = \mathbf{K}[\mathbf{I} \mid 0]\mathbf{x}_p \quad \mathbf{K} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.3}$$



Obrázek 2.1: Model dírkové kamery

Matice $\mathbf{K}$ se nazýva matice vnitřních parametrů kamery.

Obecně je bod $\mathbf{x}$ v euklidovském prostoru popsán v jiném souřadném systému, než je ten se kterým je svázána kamera. Tento souřadný systém se nejčastěji nazývá světový. Přechod do souřadného systému kamery ze světového je možno popsat následovně

$$\mathbf{x}_p = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\tilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \mathbf{x}. \tag{2.4}$$

Jako $\tilde{\mathbf{C}}$ jsme označili počátek souřadného systému kamery vyjádřený v světových souřadnicích a $\mathbf{R}$ rotaci světového souřadného systému vzhledem ke kameře. Dosadíme-li nyní rovnici 2.3 do 2.4 dostaneme vztah popisující transformaci bodu z světových souřadnic do souřadnic obrazu

$$\mathbf{x} = \mathbf{K}\mathbf{R}[\mathbf{I} \mid -\tilde{\mathbf{C}}]\mathbf{x}. \tag{2.5}$$

Parametry $\tilde{\mathbf{C}}$ a $\mathbf{R}$ se souhrně nazývají vnější parametry kamery.

2.2 Stereo kamera

2.2.1 Výpočet vzdálenosti

Stereo kamera je tvořena dvěmi nezávislými čočkami. Zpracováním obrazů z obou čoček jsme schopni získat informace o hloubce každého pixelu. Budeme uvažovat model stereo kamery, kde jsou osy obou čoček rovnoběžné (viz obrázek 2.2). Osy jsou od sebe ve vzdálenosti b. Tomuto parametru se říka báze (baseline). Z podobnosti trojúhelníků plyne

$$\frac{z}{f} = \frac{x}{xl} \tag{2.6}$$

$$\frac{z}{f} = \frac{x-b}{xr}. \tag{2.7}$$

Pokud z rovnice 2.6 vyjádříme x a dosadíme do rovnice 2.7, pak po úpravě dostaneme výsledný vztah pro vzdálenost bodu od kamery

$$z = \frac{fb}{xl - xr} = \frac{fb}{d}. \tag{2.8}$$

Hodnota d se nazývá rozdíl (disparity) a xl, xr jsou souřadnice bodu $\mathbf{p}$ v levém, resp. pravém obrazu stereokamery. Pro každý bod v levém obrazu je tedy potřeba najít odpovídající bod v obrazu pravém (tzv. stereo pár). Bez jakékoliv apriorní znalosti by to znamenalo pro každý pixel prohledat celý druhý obraz, tedy náročnost hledání $\mathcal{O}(n^2)$. Pomocí epipolární geometrie můžeme hledání zúžit na přímku a snížit tak náročnost hledání stereo páru na $\mathcal{O}(n)$. [5]

2.2.2 Chyba výpočtu vzdálenosti

Chyba výpočtu vzdálenosti se určí derivací rovnice 2.8 podle d [12]

$$\frac{\partial z}{\partial d} = \frac{z}{fb} \tag{2.9}$$

$$|\epsilon_z| = \frac{z^2}{fb}|\epsilon_d|, \tag{2.10}$$

kde ϵ_z je chyba určení vzdálenosti bodu a ϵ_d je chyba rozdílu. toto je vlastnost kamery, popřípadě algoritmu použitého ke hledání odpovídajících párů, a nabývá téměř konstantních hodnot [12]. Z rovnice 2.10 tedy plyne, že chyba je úměrná kvadrátu vzdálenosti daného bodu a přesnost stereo kamer se vzdáleností rychle klesá.

2.2.3 Epipolární geometrie

Epipolární geometrie se zabývá průnikem obrazových rovin se svazkem ploch, jejichž společná osa je báze. Uvažujme dvě dírkové kamery v obecném natočení, tedy jejich obrazové roviny nemusí být rovnoběžné (viz 2.3). Z dírkové modelu kamery plyne, že bod $\mathbf{X}$ a optické středy kamer C, C' jsou koplanární a tvoří rovinu p. V místě, kde tato rovina protíná rovinu obrazu se nachází epipolární přímka l. Hledáme-li stereo pár, pak pro známý bod $\mathbf{x}$ (tj. projekce bodu $\mathbf{X}$ do optické roviny levé kamery) musíme najít odpovídající bod $\mathbf{x}'$. Tento může ležet pouze na epipolární přímce l'. Přímka l' je uřčna epipólem e' a bodem x'. Epipól se nachází v místě, kde přímka spojující optickcká centra kamer protíná rrovinu obrazu. Bod x' se určí jako

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \tag{2.11}$$

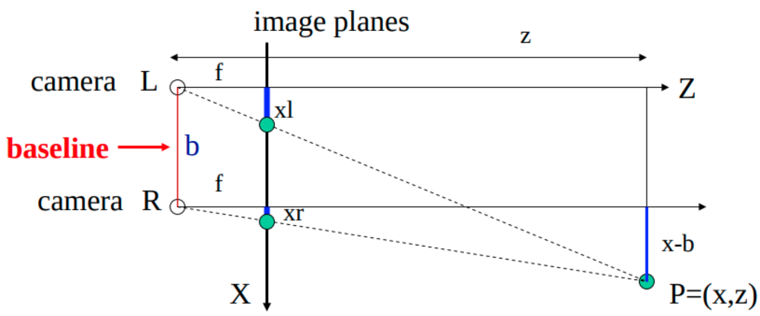
Kde $\mathbf{H}$ je homografie mezi rovinami obrazu . Toto mapování je ovlivněno pouze vzájemnou polohou kamer a jejich vnitřními parametry a nezávisí tedy na scéně. Z bodu $\mathbf{x}'$ se pak již určí epipolární přímka $\mathbf{l}'$.

$$\mathbf{l}' = \mathbf{e}' \times \mathbf{x}' \tag{2.12}$$

$$\mathbf{l}' = \mathbf{e}' \times \mathbf{H}\mathbf{x} \tag{2.13}$$

$$\mathbf{l}' = \mathbf{F}\mathbf{x} \tag{2.14}$$

Matice $\mathbf{F}$ se nazývá fundamentální a $\operatorname{rank}\mathbf{F} = 2$, tedy zobrazení reprezentované maticí F zobrazí bod na přímku. Pokud jsou obě optické osy stereo kamery rovnoběžné, pak se výpočet epipolární přímky velice zjednoduší, jelikož homografie mezi rovinami obrazu je pouze posunem ve směru báze.



Obrázek 2.2: [2]

2.3 Hledání stereo páru

Metody užívané pro hledání stereo parů se dělí na metody plošné (area based) a ty založené na porovnávání rysů (features). Při hledání stereo páru se často využívá následujících omezení, která tento proces zjednoduší. Tato však v extrémních případech nemusí platit.

- Epipolární omezení: pixel v druhém obrazu se nachází (pokud existuje) na epipolární přímce (vit. 2.2.3)
- Spojitost: Rozdílová, popř. hloubková mapa by měla být po částech spojitá.
- Jedinečnost: Každý pixel v levém obrazu má k sobě právě jeden odpovídající v obrazu pravém.
- Pořadí: Pořadí pixelů v levém obraze odpovídá pořadí v ptavém

2.3.1 Plošné metody

Využívají informaci z okolí pixelu. Tyto jsou porovnány se všemi přípustnými pozicemi v druhém obrazu a bod, jehož hodnota je nejblíže vzoru rvoří stereo pár. Mezi nejpoužívanější algoritmy patří.

Součet absolutních rozdílů

Je definována konstatní velikost okolí pixelu $\mathbf{x}(u, v)$, pro který se hledá stereo pár. Tato je následně porovnávána s body, kde se může korespondující pixel nacházet, tj. body které splnují námi používaná omezení. Pro odpovídající bod $\mathbf{x}'$ tedy platí následující vztah.

$$\mathbf{x}' = \operatorname{argmin}\left(\sum_i \sum_j |\mathbf{I}_l(u+i, v+j) - \mathbf{I}_r(u+i, v+j+d)|\right) \tag{2.15}$$

kde $\mathbf{I}(u, v)$ je intenzita příslušného pixelu, d je posun okna ve druhém obrazu a součet probíha přes celé okno, tedy přes každé i, j. Vztah byl zjednodušen uvažováním horizontálních epipolárních přímek. To odpovídá konfiguraci kamery s rovnoběžnými optickými osami.

Existuje spoustu podobných algoritmů, které pouze používají jinou metrickou funkci. Místo absolutní hodnoty například kvadrát rozdílu, nebo normalizovaný kvadrát rozdílu. Jedná se o nejrychlejší algoritmy pro hledání stereo párů, přesto však dosahují vyských přesností [13].

Cenzus

Jedná se o metodu patřící do rodiny plošných metod, které před porovnáním hodnot pixelů provedou určitou transformaci obou obrazů, sem patří nařîklad i transformace podle hodnoty (rank transform). Tato metoda opět pracuje se definovanou maskou kolem pixelu, pro který hledá stero pár. Pixely, které se nachází v šabloně se přemění na vektor jedniček a nul, podle toho zda je hodnota intenzita větší, popřípadě menší než intenzita centrálního pixelu. Stejná transformace se provede i v pravém obrazu pro každý pixel, který může doplnit stereo pár. Vektory jsou následně porovnány a je vybrán ten, jehož vektor má nejmenší Hammingovu vzdálenost. [13, 5]

2.3.2 Porovnávání rysů

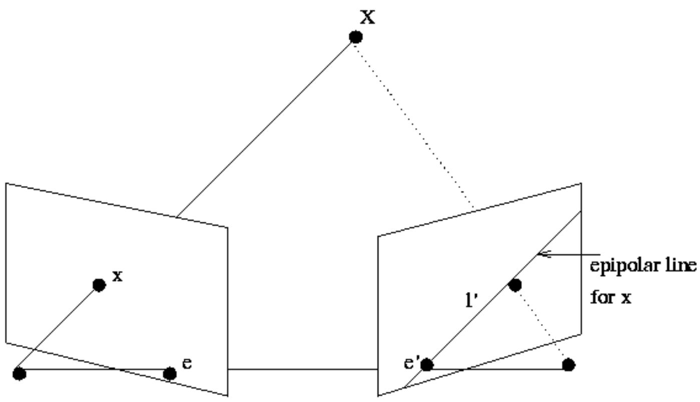
Pro každý obraz jsou vygenerovány důležité body. Tyto mohou být nepříklad hrany či rohy. Tyto jsou porovnány s analogicky vygenerovánými rysy v druhém obrazu. Tento postup se často opakuje, kdy v každém dalším kroku je rozložen každý rys na několik menších, o kterých již máme apriorní znalost jejich přibližné polohy. Aby bylo možné sestavit obraz pouze z několika rysů je nutné provést časově náročné předzpracování obrazu za účelem najití vhodných rysů a po nalezení korespondujících rysů provést zpětnou rekonstrukci obrazu. [5]

Pro porovnání rysů mezi levým a pravým obrazem se využívá algoritmů jako je například SIFT (Scale Invariant Feature Transform), SURF (Seeded-Up Robust Features) nebo BRIEF(Binary Robust Independent Elementary Features).

2.4 Intel Realsense D435

Intel® Realsense™ D435 je širokoúhlá stereo kamera, která se skládá z RGB kamery, infračerveného (infra red)projektoru, a dvou infračervených kamer jejichž data jsou zpracovává přímo v čipu kamery procesor. Výstupem z kamery je tedy barevný obraz (RGB) a vzdálenost jednotlivých pixelů neboli hloubková mapa.

Dvě infračervené kamery jsou v konfiguraci s rovnoběžnými optickými osami ve vzdálenosti 50 mm. Infračervený projektor promítá na scénu statický obrazec, který se nachází mimo viditelné spektrum a není tedy zachycen RGB kamerou. Tento obrazec slouží k najití stereo párů, zejména pak u objektů s řídkou texturou.



Obrázek 2.3: Placeholder

Obrázek 2.4: Fotografie kamery Intel® Realsense$^{\mathrm{TM}}$ D435 [1]

Kamera je vybavena specializovaným procsorem Vision Processor D4 pro výpočet hloubkové mapy. Tento je schopen při rozlišení hloubkové kamery 848x480 zpracovávat až 90 snímků za sekundu (FPS), toto je vykoupeno sníženou přesností [12] a pro maximální přesnost je vhodné snížit frekvenci na 30 snímků za sekundu. Procesor má stejně jako celá kamera velice nízký odběr. Celá kamera včetně IR projektoru má odběr menší než jeden watt a je tedy vhodná pro mobilní aplikace, kde je velikost baterie a spotřeba energie omezujícím faktorem. [3]

Pro hledání stereo párů je využit cenzus algoritmus popsaný v 2.3.1 s maskou velikosti 7×7. Výsledky jsou následně ověřovány sadou filtrů, které měří důvěryhodnost shody. Podle nastaveného limitu pak pro tento pixel bud vygenerují záznam v hloubkové mapě, nebo pixel zůstane nevyplněn. Výsledky kamery pro různé hodnoty limitu jsou vidět v tabulce.

Procesor kamery není schopen určit hloubku bodu, pokud se objekt nachází příliš blízko. Konkrétní hodnoty je možno vidět v tabulce 2.6. Maximální měřitelná vzdálenost je za ideálních podmínek až 40 metrů [12]. S rostoucí vzdáleností se kamera chová podle rovnice 2.10, toto chvání není patrné, pokud kamera operuje s objekty, které se nachází na obou hranicích měřitelné vzdálenosti. Ilustrace výstupu kamery, pokud se objekt nachází ve vzdálenosti meší, než specifikuje tabulka 2.5 je vidět na obrázku 2.7b.

Preset	FPR	$r_{\rho=95\%}$	$\max(\sigma_x, \sigma_y) > \sigma_z$
Off	91.3%	7.0 m	6.1 m
Low	19.8%	6.9 m	6.6 m
Medium	5.8%	5.8 m	6.7 m
High	0.5%	4.2 m	6.8 m

Obrázek 2.5: Postupně zleva: hodnota limitu, počet falešných shod pokud je scéna blíže, než je minimální vzdálenost detekce kamerou. Vzdálenost, kdy vyplněnost hloubkové mapy klesne pod 95% a vzdálenost při které je směrodatná odchylka ve směru osy z menší, než ve směru x a y [12]

Resolution	D400/D410/D415	D420/D430
	Min-Z (mm)	Min-Z (mm)
1280x720	450	280
848X480	310	195
640x480	310	175
640x360	240	150
480x270	180	120
424x240	160	105

Obrázek 2.6: Tabulka minimální detekovatelné hloubky [3]

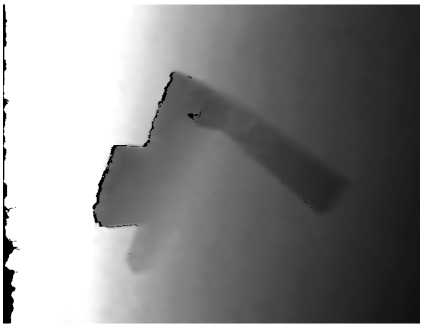
2.5 Metody v detekci objektů z hloubkových dat

Existuje spostu růyných metod pro detekci objektů z hloubkových dat. Tyto se liší podle různorodosti detekovaných objektů, požadavků na čas výpočtu, možnosti získání ténovacích dat nebo formátem vstupních dat. Obvyklím formátem je mračno bodů, neboli point cloud, což je nestrukturovaná množina bodů v trojrozměrném prostoru. Zejména v posledních letech s nástupem hloubkových kamer jako je například Microsoft Kinect, Asus Xtion nebo námi používaný Intel RealSense, se čím dál více používájí data ve formátu RGB-D obrazu, tento se dá pomocí vztahu 2.2 převést na point cloud. Výstupní data z hloubkové kamery bývají většinou oproti LIDARu méně přesné. Výhodami naopak jsou: Vysoká vzorkovací frekvence, velké množstvý výstupních bodů a cena.

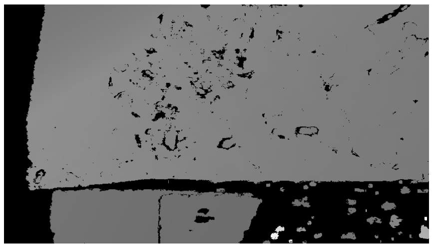
2.5.1 Klasické metody

Určení normálového vektoru plochy z nestrukturovaných dat

Téměř všehny metody detekce z objektů se skládá z více kroků a jedním z nich většinou bývá určení normálového vektoru plochy. Obvykle se pak jedná o plochu reprezentující zem, popřípadě jinou podložku, popřípadě plochy reprezentující stěny místnosti. Přístupy k hledání ploch v obrazu se výrazně liší, uvedeme příklady některých z nich.



(a) : Snímek bez výrazných výpadků dat hloubkové mapy



(b) : Snímek s chybějícími hloubkovými daty

Obrázek 2.7: Výstupní data kamery Realsense, hloubková data byla převedena do odstínů šedé

- Použití hrubé síly (brute force). Postupně jsou vyzkoušeny všechny možnosti, popřípadě část z nich a vybere se nejlépe vyhovující. Využívá se zde zejména algoritmů jako je RANSAC, popřípadě jeho modifikace. Tato metoda je vhodná zejména pokud jsou data zatížena silným šumem, jelikož při správném počtu iterací téměř vždy vrátí správný výsledek, toto je však vykoupeno vysokou časovou náročností.[8, 4] Tato se dá snížit vhodný předzpracováním dat. Například segmentací na menší celky a určením plochy pouze pro tyto segmenty [16]

- Výpočet normálového vektoru v každém bodě, kde je normálový vektor kolmý na plochu reprezentující okolí bodu, toto je tvořeno body nacházejícími se v okruhu o poloměru r [19] popřípadě k nejbližšími body[9, 18]. Poté je provedena segmentace pomocí metody rostoucích oblastí viz. sekce 2.5.2. Výstupem jsou tedy skupiny bodů, které reprezentují jednotlivé plochy. V některých případech je ještě použit RANSAC na již nalezené segmenty, aby byly vyloučeny body, které leží mimo plochu přesto, že jejich normálový vektor dané ploše odpovídá [14].

- Jsou nalezeny body, které patří do dané roviny, těmito je poté pomocí PCA proložena rovina [22]. Tato metoda je náchylná na body ležící mimo rovinu (outliers), jelikož rovina je proložena body ve smyslu nejměnších čtverců a jediný bod ležící ve velké vzdálenosti od roviny může způsobit velké nepřesnosti.

2.5.2 Segmentace obrazu

Důležitým krokem pro porozumění obrazu a pro detekci objektů je segmentace obrazů, tj. rozdělení vstupních dat na skupiny. Rozdělujeme dva typy segmentaco a to sémantickou, která sjednocuje body, které reprezentují danou skupinu objektů (například kostky ležící na zemi). Druhým typem je segmentace jednotlivých instancí, tedy skupina objektů je rozdělena na jednotlivé instance dané skupiny.

Přístup k segmentaci se liší podle dané aplikce. Nejpoužívanějšími metodami jsou tyto:

- Metoda prahování. Body $\mathbf{p}(x, y)$ jsou rozděleny do m skupin podle toho, zda jejich vlastnost vlastnost V, což je například vzdálenost bodu od kamery nebo od plochy, dosahuje stanovené hodnoty T_m, tato můžeme být pevně určená, nebo se může dynamicky měnit a to v závislosti na okolních bodech, nebo v závislosti na pozici v obrazu, tj. $T(x, y)$. Pro prahovací metodu tedy platí následující vztah

$$\left\{ \begin{matrix} \mathbf{p} \in m \text{ iff } V(\mathbf{p}) > T_m \\ \mathbf{p} \in n \text{ iff } V(\mathbf{p}) > T_n \\ \mathbf{p} \in \emptyset \text{ iff } V(\mathbf{p}) \leq T_1 \end{matrix} \right\} \tag{2.16}$$

- Metoda detekce hran. V obrazu jsou detekovány prudké změny pozorované vlastnosti V bodů obrazu. Pokud první derivace bodu $\mathbf{p}$ dosáhne dané hodnoty je $\mathbf{p}$ registrován jako hrana. Povýpočtu bývá provedena úprava těchto hran, kterou je nepříklad filtrování, popřípadě spojení některých sousedních bodů. Nakonec jsou v obrazu identifikovány segmenty bodů, které jsou od zbytku odděleny hranou. K výpočtu se využívá například Sobelova oprátoru, který byl popsán v sekci 1.3 nebo Cannyho operátor.

- Metoda rostoucí oblasti. Tato se dělí na dva typy: S počátečním bodem (seed) a bez počátečního bodu.

  Je zvolen jeden, popřípadě několik počátečního bodů. K těmto se postupně přidávají sousední body, pokud splňují předem definované podmínky. Těmito je například maximální odchylka normálových vektorů bodů od normálového vektoru seedu, nebo rozdíl velikost některé souřadnice. Metoda bez počátečního bodu pak rozdělí všechny body do skupin, jejichž body spolu sousedí a zároveň mají podobné vlastnoti. Jedná se o algoritmus, který pracuje s konstatní časovou náročností, která odpovídá $\mathcal{O}(2n)$, kde n je počet pixelů obrazu.

- Metoda rozdělovaní a slučování. Obraz je postupně rozdělen na části, které mají podobnou charakteristiku. Následně jsou seousední regiony, které jsou si podobné sloučeny do jedoho. Pro reprezentaci regionů se obvykle využívá quadtree.

  Pokud o je obraz a V daná vlastnost jednotlivého bodu p a skupina bodů má vlastnost V, pokud tuto má každý bod skupiny pak lze metodu rozdělování a slučování popsat následovně.[11]

  - Region R_1 je roven o
  - Pokud platí $V(R_i) = False$, pak je region rezdělen na několik menších
  - Pokud platí $V(R_i) = True$, pak je R_i slučen se všemi sousedními regiony R_j, přičemž musí platit $V(R_i \cup R_j)$. Tento krok se opakuje dokud je možné některý region sloučit.

- Metoda shlukování (clustering). Obraz je rozdělen do clustrů, kde body v každém clustru mají podobné vlastnosti, resp. jejich vlastnosti jsou si v rámci obraz nejbližší. Vlastnost v tomto případě bývá nejčastěji euklidovská vzdálenost. Mezi nejpopulárnější algoritmy patří tyto [1]
  - K-means algoritmus. V tomto algoritmu je předem nutné znát počet clustrů m, do kterých budou body rozděleny. Následně se náhodně vybere m bodů, které představují střed clustru. Následně je pro každý bod spočítána vzdálenost od jednotlivých středů. Bod je přiřazen do clustru, jehož vzdálenost středu je nejmenší. Následuje přepočítaní středů a postup se opakuje. Ve chvíli kdy již nedochází ke změnám mezi jednotlivými clustery je zaznamenán součet rozptylů bodů v rámci jednotlivých clusterů a celý proces počínaje náhodnou volbou středů se n-krát opakuje. Nakonec je vybrám výsledek, který má nejmenší roptyl.
  - Mean-shift (posun průměru) algoritmus. Je předem definován poloměr kruhového okna r. Následně je náhodně určeno (popřípadě podle předem definované masky rozmístěno) n bodů $\mathbf{p}$. V každé iteraci se spočítá střed bodů $\mathbf{t}$, pro které platí $||\mathbf{p} - \mathbf{t}|| < r$ a tento naradí bod $\mathbf{p}$, tento postup se opakuje, dokud body konvergují. Výsledná poloha bodů $\mathbf{p}_i$ pak představuje sřed jednotlivých clustrů.
  - DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algoritmus. Všechny body obrazu jsou označeny jako nenavštívené. Náhodně je vybrán nenavštívený bod $\mathbf{a}$ a pokud se v jeho okolí o velikosti ϵ nachází minimální předem zvolený počet bodů, pak je celé okolí bodu $\mathbf{p}$ přídáno do clusteru a postup se opakuje pro přidané body. Pokud již nelze žadný další bod přidat zvolí se další nenavštívený bod a proces se opakuje pro další cluster. Toto probíhá dokud existují nanavštívené body.

Detekce objektů

2.6 Navržené řešení

Navržené řešení se skládá ze 3 částí a to v souladu se sekcí 2.5, tedy nalezení normálového vektoru podložky, segmentace obrazu na jednotlivé cihly a následně nelezení ohraničujících boxů pro jednotlivé cihly. Pro všechny části řešení bylo navrhnuto více postupů a jejich výsledky jsou porovnány v sekci 2.8. Ve všech částech se výchazí ze zadání, které specifikuje, že na zemi nemůže být krom cihel žadný jiný objekt podobných rozměrů a velikost cihel je předem známa, krom jejich délky, která může nabývat ruzných rozměrů. Cihly na sebe mohou být naskládany v libovolném počtu vrstev a mohou se objevit v libovolném natočení a navzájem se dotýkat. V některých částech byl zaveden předpodklad doteku dvou cihel pouze po celé délce odpovídajících stěn za účelem nabídnout rychlejší alternativu detekce.

[1] Princip algoritmů bude vysvětlen při shlukování podle euklidovské vzdálenosti

2.6.1 Segmentace obrazu

Přístup založený na výšce

Tento postup vychází z předpokladu, že na zemi se krom cihel nenachází jiné předměty. Scénu tedy tvoří pouze zem a na ní umístěné cihly, které mají předem známé rozměry. Začneme určením normálového vektoru roviny reprezentující zem, jelikož z jeho znalosti můžeme najít body ležící nad zemí, tj. body představující cihly. Vytvoříme mřížku bodů, které jsou v místech, kde předpokládáme přesný výstup kamery. U těchto ověříme, zda pro ně byla kameoru vygenerována hloubka, dále ověřme, zda bod leží v maximální vzdálenosti $4\,m$ jelikož podle [12] chyba stereo kamery RealSense D435 nad tuto vdálenost začíná být markantní a správné určení normálového vektoru je zásadní pro všechny ostatní kroky. Množinu bodů $M = \mathbf{a}_1 \ldots \mathbf{a}_m$ které splňují výše uvedná kritéria proložíme rovinou pomocí PCA algoritmu a dostaneme tedy normálový vektor $\mathbf{n}$ plochy p minimalizující kvadrát vzdáleností od bodů tato je posunuta mimo počátek o d.

$$\bar{\mathbf{a}} = \frac{1}{m}(\mathbf{a}_1 \ldots \mathbf{a}_m) \tag{2.17}$$

$$d = \mathbf{n} \cdot \bar{\mathbf{a}} = \mathbf{n}^T \bar{\mathbf{a}} \tag{2.18}$$

$$p : n_1 x + n_2 y + n_3 z - d = 0 \tag{2.19}$$

Každý bod $\mathbf{a}$ je buď součástí země, nebo součastí cihly. Je-li v množině M k bodů, které jsou součastí země G, pak zbylých $m - k$ bodů musí být součastí mračna bodů reprezentující cihlu C. Tyto body se vždy nachází ve výšce z [2]nad zemí a tedy platí, pro každý bod $\mathbf{a}$

$$n_1 a_1 + n_2 a_2 + n_3 a_3 + \epsilon > d \;\rightarrow\; \mathbf{a} \in C \tag{2.20}$$

$$n_1 a_1 + n_2 a_2 + n_3 a_3 + \epsilon \leq d \;\rightarrow\; \mathbf{a} \in G \tag{2.21}$$

Kde ϵ je experimentálně určená hodnota, která ošetří případy, kde $k = m$, tj. všechny body reprezentují zem. V tomto případě čast bodů bude ležet nad plochou p. To jest dáno šumem dat a nepřesností kamery. Přidáním parametru ϵ je zajištěno, že každý bod reprezentující zem bude registrován i při zašuměných datech.

Po výběru bodů z mřížky podle výše uvedených kriterií dostaneme k bodů, které reprezentují zem a nacházejí se v místech, kde je přesnost kamery v rámci jejich možností nejvyšší. Tyto body by se daly znovu proložit plochou a dostat tak aproximaci natočení země pomocí normálového vektoru $\mathbf{n}$. Vzhledem k důležitosti přesnosti určení $\mathbf{n}$ získáme další body reprezentující zem pomocí algoritmu 1. Tyto pody opět pomocí PCA proložíme přímkou a dostaneme dobrou aproximaci normálového vektoru země. Na obrázku je vidět výsledek algoritmu 1.

Nyní se vypočítá výška každého bodu nad rovinou a podle této hodnoty se přidělí hodnota 0 až 3, kde číslo reprezentuje očekávaný počet na sobě naskládaných cihel. Vizualizace tohoto výsledku je vidět na obrázku 2.9

[2]Osa z směřuje směrem z kamery a tedy body nacházající se nad zemí mají menší hodnotu z

Algorithm 1 Epadndování bodů

```
Stack ← ∅
G ← ∅
for a in M do
  Stack.push((a))
  z ← height of a
  while not Stack.empt() do
    t ← Stack.pop()
    G ← G ∪ t
    V ← V ∪ p
    for n in neighbours of t do
      zn ← height of n
      if zn ∈ [z − δ, z + δ] and not n ∈ G then
        stack.push(n)
      end if
    end for
  end while
end for
```

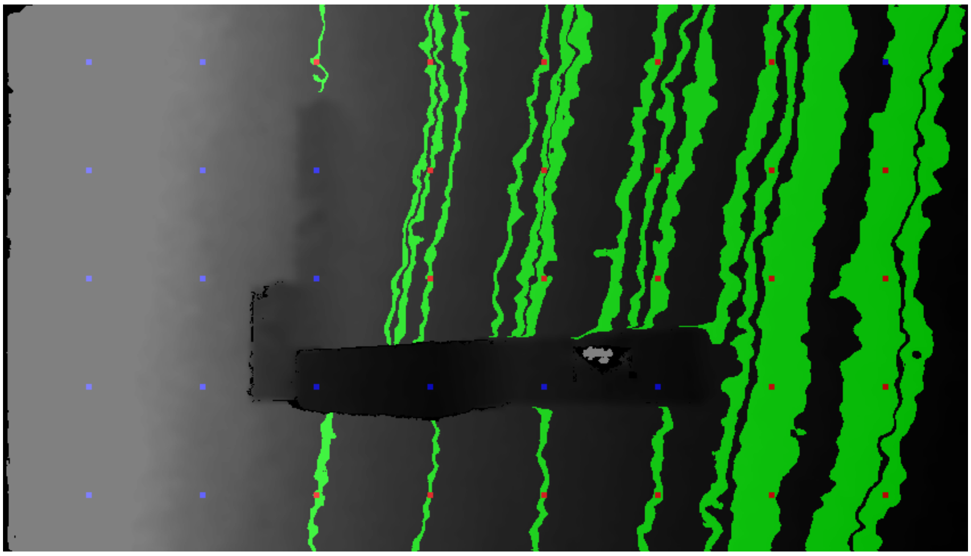
Přístup založený na změně výšky

Velice populární postup při segmentaci obrazu je založena na normálovém vektoru, tento je vypočten pro každý bod obrazu a následně jsou body, jejiž normálový vektor má podobný směr, popřípadě i velikost, sloučeny do větších celků. V našem případe je vrchní stěna cihly parelelní se zemí a má tedy stejný normálový vektor. Nachází se však v jiné výšce. Pomočí výpočtu gradientu, který budeme realizovat pomocí Sobelova operátoru aplikovaného na výšku bodů, tedy spočítáme derivaci výšky, tato by měla být nejvyšší v oblasti přechodu země - cihla. Následně pak sloučíme body s hodnotou gradientu výšky, která si liší v rámci skupiny maximálně o δ. K tomu bylo využito upraveného CCL (Connected-component labeling) algoritmu.

Tato metoda při správném odladění parametru δ funguje velice přesně. Problémem je však robustnost. Hodnoty parametru δ fungující na cihly v blízkosti kamery, nedtekují cihly ve vzdálenosti řadově $2\,m$ a více a naopak. Toto je způsobeno jednak šumem kamery, který podle rovnice 2.10 roste kvadraticky a zejména pak zkreslením vzdálenější hrany cihly kamerou, viz. obrázek 2.10. Kamera zde špatně nachází stereo páry a generuje mračna bodů, které ve skutečnosti neexistují. Tyto zmírňují přechod cihla-zem a snižují tak hodnotu gradientu. Toto chování je dobře vidět na obrázku 2.11.

2.6.2 Detekce objektů ze segmentovaného obrazu

Jako při segmentaci obrazu, tak i zde bylo vyzkoušeno několik algoritmů. Některé z nich byly nastaveny na detekci za zjednodušujících podmínek a nabízí tak nižší výpočetní náročnost při stejné přesnosti detekce. Vstupem všech prezentovaných algoritmů bude segmentovaný obraz jako je například



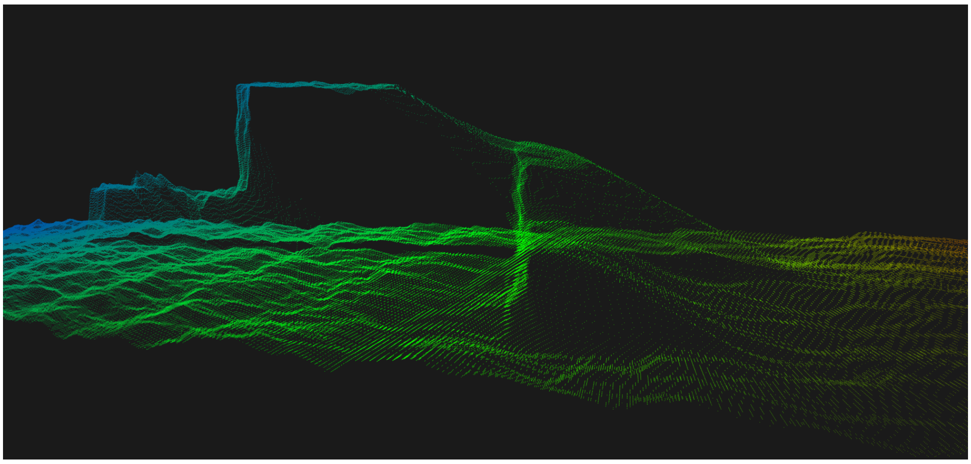
Obrázek 2.8: Červenou barvou jsou znázorněny body, ze kterých probýhal algoritmus 1, modré body byly podle výše uvedených kritérií vyřazeny a zeleně jsou znázorněny výsledné body reprezentující zem

obrázek 2.9, pro tuto segmentaci jsme využili algoritmus 2.6.1.

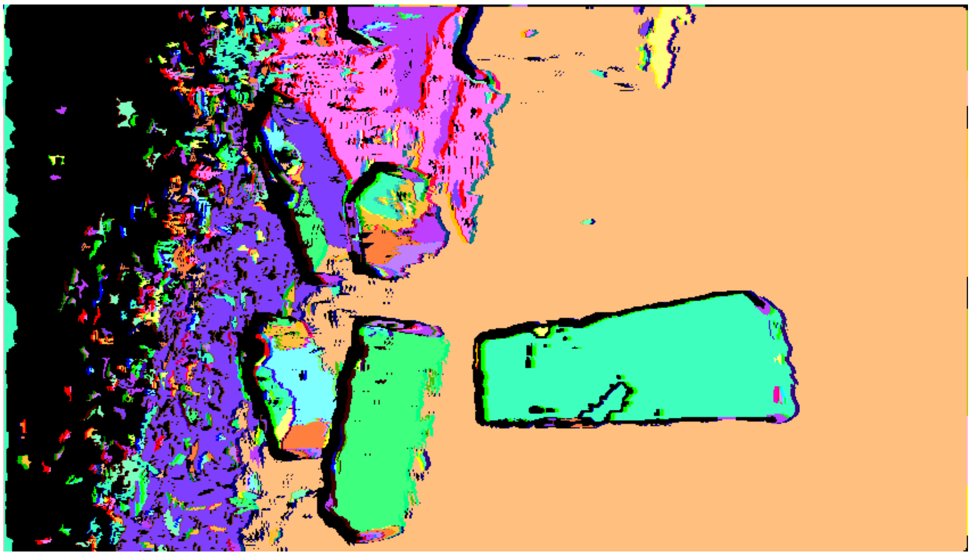
2.7 Výsledky

2.8 Závěr

Obrázek 2.9: Vzdálenost boudů od roviny zobrazena jako počet na sobě lžících cihel



Obrázek 2.10: Výstup kamery - mračno bodů zobrazující skupinu cihel. V pravé části je vidět zkreslení hrany způsobující problém při segmentaci



Obrázek 2.11: Výsledek segmentace obrazu podle velikosti derivace výšky.

Literatura

[1] *The Intel® RealSense™ Depth Camera D435.*

[2] Stereo and 3d vision.

[3] *Intel® RealSense™ D400 Series Product Family*, 2019.

[4] Panagiotis-Alexandros Bokaris, Damien Muselet, and Alain Trémeau. 3D reconstruction of indoor scenes using a single RGB-D image. In *12th International Conference on Computer Vision Theory and Applications (VISAPP 2017)*, Porto, Portugal, February 2017.

[5] Myron Z Brown, Darius Burschka, and Gregory D Hager. Advances in computational stereo. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):993–1008, 2003.

[6] R. V. Chadnov and A. V. Skvortsov. Convex hull algorithms review. In *Proceedings. The 8th Russian-Korean International Symposium on Science and Technology, 2004. KORUS 2004.*, volume 2, pages 112–115 vol. 2, 2004.

[7] Timothy M Chan. Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete & Computational Geometry*, 16(4):361–368, 1996.

[8] Orazio Gallo, Roberto Manduchi, and Abbas Rafii. Cc-ransac: Fitting planes in the presence of multiple surfaces in range data. *Pattern Recognition Letters*, 32(3):403–410, 2011.

[9] Dirk Holz, Stefan Holzer, Radu Bogdan Rusu, and Sven Behnke. Real-time plane segmentation using rgb-d cameras. In *Robot Soccer World Cup*, pages 306–317. Springer, 2011.

[10] Walker James W. Angle Between Vectors, nov 2014.

[11] Dilpreet Kaur and Yadwinder Kaur. Various image segmentation techniques: a review. *International Journal of Computer Science and Mobile Computing*, 3(5):809–814, 2014.

[12] Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. Intel realsense stereoscopic depth cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–10, 2017.

[13] Annika Kuhl. Comparison of stereo matching algorithms for mobile robots. *The University of Western Australia Faculty of Engineering, Computing and Mathematics*, 2005.

[14] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *2011 IEEE international conference on robotics and automation*, pages 1817–1824. IEEE, 2011.

[15] David Pichardie and Yves Bertot. Formalizing convex hull algorithms. In *International Conference on Theorem Proving in Higher Order Logics*, pages 346–361. Springer, 2001.

[16] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Close-range scene segmentation and reconstruction of 3d point cloud maps for mobile manipulation in domestic environments. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1–6. IEEE, 2009.

[17] Godfried T Toussaint. Complexity, convexity, and unimodality. *International journal of computer & information sciences*, 13(3):197–217, 1984.

[18] Alexander JB Trevor, Suat Gedikli, Radu B Rusu, and Henrik I Christensen. Efficient organized point cloud segmentation with connected components. *Semantic Perception Mapping and Exploration (SPME)*, 2013.

[19] Jiefei Wang, Matthew Garratt, and Sreenatha Anavatti. Dominant plane detection using a rgb-d camera for autonomous navigation. In *2015 6th International Conference on Automation, Robotics and Applications (ICARA)*, pages 456–460. IEEE, 2015.

[20] Serge Weisstein Eric W., Belonie. Rodrigues' rotation formula. Visited on 8/04/20.

[21] Tomáš Werner. *Optimalizace*. České vysoké učení technické, 11.2.2020 edition, 2020.

[22] Lizhi Zhang, Diansheng Chen, and Weihui Liu. Fast plane segmentation with line primitives for rgb-d sensor. *International Journal of Advanced Robotic Systems*, 13(6):1729881416665846, 2016.