ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

KATEDRA APLIKOVANÉ ELEKTRONIKY A TELEKOMUNIKACÍ

BAKALÁŘSKÁ PRÁCE

Cluster analýza pro pixelové detektory částic

Autor práce: Jan Komín
Vedoucí práce: Ing. Jan Broulím Ph.D.

Plzeň 2019

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: **2018/2019**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Jan KOMÍN**
Osobní číslo:	**E15B0010P**
Studijní program:	**B2612 Elektrotechnika a informatika**
Studijní obor:	**Elektronika a telekomunikace**
Název tématu:	**Cluster analýza pro pixelové detektory částic**
Zadávající katedra:	**Katedra aplikované elektroniky a telekomunikací**

Zásady pro vypracování:

1. Proveďte rešerži metod cluster analýzy.
2. Diskutujte vhodnost metod pro použití v částicových detektorech.
3. Vybrané metody implementujte na zvolené platformě a vhodně prezentujte.
4. Proveďte měření rychlosti zpracování a diskutujte vhodnost použití pro online měření.

Rozsah grafických prací: **podle doporučení vedoucího**

Rozsah kvalifikační práce: **30 - 40 stran**

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **Bailey, Ken: Numerical Taxonomy and Cluster Analysis. Typologies and Taxonomies. 1994**
2. **Everitt, B.,S.; Landau, S.; Leese, M.: Cluster Analysis. 2001**

Vedoucí bakalářské práce: **Ing. Jan Broulím**
Regionální inovační centrum elektrotechniky

Datum zadání bakalářské práce: **5. října 2018**

Termín odevzdání bakalářské práce: **13. června 2019**

Prof. Ing. Zdeněk Peroutka, Ph.D.
děkan

L.S.

Doc. Dr. Ing. Vjačeslav Georgiev
vedoucí katedry

V Plzni dne 5. října 2018

Abstrakt

Komín, Jan. Cluster analýza pro pixelové detektory částic. Západočeská univerzita v Plzni, Fakulta elektrotechnická, Katedra aplikované elektroniky a telekomunikací, 2019, 49 s., vedoucí práce: Ing. Jan Broulím, Ph.D.

Bakalářská práce se zabývá Cluster analýzou pro pixelové detektory částic. Bakalářská práce je rozdělena na dvě části. První část je věnována obecným metodám a metodám, které se využívají právě pro pixelové detektory. Druhá část práce je věnována vlastnímu algoritmu. Algoritmus je psán v programu Matlab.

Klíčová slova

Cluster analýza, cluster, K-Means, K-Medoids, Medipix2, Timepix

Abstract

Komín, Jan. Cluster analysis for pixel detectors. University of West Bohemia in Pilsen, Faculty of electrical engineering, Department of applied electronics and telecommunications, 2019, 49 p., head: Ing. Jan Broulím, Ph.D.

The bachelor thesis deals with Cluster analysis for pixel detectors. The bachelor thesis is divided into two parts. The first part is dedicated to general methods and methods that are used for pixel detectors.The second part is dedicated to the algorithm itself. The algorithm is written in Matlab.

Keywords

Cluster analysis, cluster, K-Means, K-Medoids, Medipix2, Timepix

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této bakalářské práce.

Dále prohlašuji, že veškerý software, použitý při řešení této bakalářské práce, je legální.

. . . . . .

Podpis

V Plzni, dne . . . . . Jan Komín

Poděkování

Chtěl bych poděkovat Ing. Janu Broulímovi, Ph.D. za odborné vedení práce a cenné rady, které mi pomohly ke zpracování bakalářské práce.

Obsah

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

BW Black and White image (binary image). Černobílý obraz (binární obraz).

CC Connected Components. Připojené komponenty.

GPU Graphic Processing Unit. Grafický procesor.

lin linearity. Linearita.

PAM Partition Around Medoids. Rozdělení kolem medoidů (těžišť).

ToA Time of Arrival. Čas výskytu.

ToT Time over Threshold. Doba nad prahovou hodnotou.

1 Úvod

Cílem bakalářské práce je vysvětlit pojem Cluster analýza, popsat metody, které se v této analýze používají a popsat vybrané metody, které se přímo využívají v částicových detektorech. Cluster analýza se do českého jazyka překládá jako tzv. Shluková analýza. To znamená, že se zde bavíme o určitých skupinách, které jsou definovány různými typy shluků. Shluky jsou tvořeny pixely, které lze zařadit do předdefinovaných skupin.

V druhé části bakalářské práce se zabývám daty, která byla reálně naměřena. Data se nacházejí v textovém souboru, který se musí převést na matici o rozměrech 256×256. Převedení i další zpracování řeším v programu Matlab, kde vhodným algoritmem procházím matici a hledám vlastnosti clusterů jako například jejich centroid (těžiště), souřadnice a počet pixelů.

V závěru vyhodnocuji použitý algoritmus a výsledky, ke kterým jsem dospěl. Také vyhodnocuji, jak dlouho trvalo vytvořenému algoritmu prohledat matici a zda tento algoritmus lze použít pro online řešení.

2 Cluster analýza

2.1 Popis problematiky Cluster analýzy

Každý jedinec multivariačního (vícerozměrného) vzorku může být reprezentován bodem ve vícerozměrném euklidovském prostoru. Cluster analýza se pokouší seskupit tyto body do tzv. disjunktních sad (tj. do sad, které nemají společný prvek) a u kterých se očekává, že budou odpovídat výrazným vlastnostem vzorku [4].

Slovo cluster (klastr) můžeme definovat jako sbírku datových objektů [2].

Různé metody Cluster analýzy stejného vzorku mohou předpokládat různé geometrické rozložení bodů, nebo mohou používat různá kritéria shlukování, nebo se mohou lišit v obou ohledech.

Cluster analýza se stala populární mezi některými systematisty zabývajícími se vytvářením klasifikací. Existuje mnoho technik této analýzy a je obtížné posoudit jejich relativní výhody a nevýhody, protože cluster není dobře definovaný koncept. Může se stát, že je třeba definovat několik různých typů clusteru, a v takovém případě by měl vyšetřovatel zvážit, která definice splňuje jeho požadavky. Mohl by být nalezen více než jeden vhodný algoritmus, ale všechny tyto algortimy by měly dát stejný výsledek s výjimkou, kdy údaje připouští více než jednu sadu clusterů v souladu s definicí. Někdy je vhodnější používat jednoduché algoritmy, které poskytují pouze přibližné výsledky. Některé algoritmy zakrývají formu clusterů, které najdou a je naším cílem zde vystavit podkladovou strukturu clusteru několika metodami [4].

2.2 Příklady Clustering aplikací

Marketing

Pomáhá obchodníkům objevovat odlišné skupiny ve svých zákaznických základkách a poté pomocí těchto poznatků dopomoci k vývoji marketingových programů.

Využívání půdy

Cluster analýza slouží i pro identifikaci půdních oblastí. Porovnává, jak a kde se ve světě zemská půda využívá.

Plánování města

V plánování měst pomáhá určit skupiny domů podle typu domů, hodnoty a jejich geografické polohy [2].

2.3 Měření kvality shlukování

Porovnává se metrika odlišnosti a podobnosti. Podobnost se vyjadřuje jako funkce, která se označuje $d(i, j)$ [2].

3 Metody obecně

Tři algoritmy, které popsali Sokal a Michener [1958], Edwards a Cavalli-Sforza [1965], Wiilliam a Lambert [1959] jsou označované jako Metody I, II a III. Tyto algoritmy byly vybrány, protože mají hodně společného. Nové algoritmy neumožňují analýzu, protože studie paralelních modelů odhaluje vlastnosti definic clusteru, které by mohly být jiné, než se očekává.

Většina metod Cluster analýzy pracuje na multivariačním vzorku $\boldsymbol{N}$ jedinců s pozorováním na stejných $\boldsymbol{v}$ variacích pro každého jednotlivce. Některé metody vyžadují, aby všechny variace byly dichotomické (dvojčlenné) s hodnotami 0 až 1, což indikuje přítomnost a nepřítomnost znaků. Jiné vyžadují, aby všechny variace byly kvantitativní (tj. měřeny na stupnici). V praxi se vyskytují a mohou být zahrnuty i více hodnotné kvalitativní variace. Je výhodné si představit každého jednotlivce, jak je reprezentován bodem P, (r = 1, 2, ..., N) v mnohorozměrném euklidovském prostoru. Existuje mnoho způsobů, jak získat takové zastoupení. Například, jsou-li všechny variace kvantitativní (a možná i když jsou dichotomické), mohou být jejich hodnoty považovány za souřadnice P souvisejících se sadou pravoúhlých os. To je model předpokládaný jako základ pro analýzu hlavních složek. Jednotlivé metody jsou popsány v následujících podsekcích a jejich podrobnější popis je rozebrán v článku [4].

3.1 Metoda I (Sokal a Michener)

Metoda váženého středního páru Sokal a Michener byla původně aplikována na entomologický problém (tj. problém v oblasti zoologie zabývající se studiem hmyzu) [1958]. Nedávný popis dal Sokal a Sneath [1963], kteří jej doporučili jako nejlepší ze třídy bežně používaných metod Cluster analýzy.

Metoda pracuje na matici podobnosti $\boldsymbol{N} \times \boldsymbol{N}$, ale původně byly použity formální korelace (znázorňuje statickou závislost dvou kvantitativních veličin neboli měří vzájemný vztah dvou proměnných) mezi jednotlivými páry jedinců. Algortimus sdružuje ty dva jedince, $\boldsymbol{i}$ a $\boldsymbol{j}$, kteří mají nejvyšší podobnost a nahrazuje sloupce (a řádky) $\boldsymbol{i}$ a $\boldsymbol{j}$ jedním sloupcem s vhodně zvolenými „průměrnými“ podobnostními koeficienty. Pro nahrazení

korelace jsou použity vzorce Spearman, které udávají korelace mezi součtem standardizovaných proměnných a pro podobnosti se odebírají prostředky prvků v řádcích (a sloupcích) $\boldsymbol{i}$ a $\boldsymbol{j}$. Tento proces se pak opakuje na nové $(N-1)$ matici řádků, kdy buď dva noví jedinci mají nejvyšší podobnost a tvoří nový pár, nebo stávající dvojice se spojí s dalším jednotlivcem, aby vytvořila cluster tří. Proces pokračuje dvojicí jedinců a seskupení dříve spojených jedinců.

3.1.1 Komentáře k Metodě I

Jedná se o aglomerační (seskupující) metodu, která vytváří clustery od jednotlivců. Spearmanové vzorce porovnávají dvě sady součtů změn u kterých jsou jednotlivé změny standardizovány tak, aby měly jednotkovou odchylku. Aby korelace mezi jednotlivci byly nezávislé na stupnicích měření změn, je každá pravá variace nejprve standardizována dělením podle její standardní chyby. Proces korelace jednotlivců účinně ničí tuto normalizaci, protože zaznamenání pro každého jednotlivce je implicitně restandardizováno dělením standardní chyby všech jeho standardizovaných zaznamenání a pak odečtením průměrného restandardizovaného zaznamenání.

3.1.2 Hodnocení podobnosti mezi skupinami

V geometrickém výkladu předpokládáme, že pokud S_{ij} je podobnost mezi jednotlivci $\boldsymbol{i}$ a $\boldsymbol{j}$, pak vzdálenost mezi jejich bodovými reprezentacemi P_i a P_j je $[2 \cdot (1 - S_{ij})]^{\frac{1}{2}}$.

Byly vybrány jiné monotonické transformace rozdílu, ale tato volba zjednodušuje algebru a vede k přímému porovnání s postupem průměrování podobností. Nahrazování podobností korelací nevytváří problém [4].

3.2 Metoda II (Edwards a Cavalli-Sforza)

Metoda II navrhuje rozdělení bodů do dvou sad tak, aby součet čtverců vzdáleností mezi množinami byl maximální. To definuje, co znamená cluster a vyžaduje pouze algoritmus k nalezení dvou sad s požadovanou vlastností. Protože celkový součet čtverců je konstantou pro daný vzorek, maximalizace součtů čtverců mezi množinami je ekvivalentní minimalizací součtu čtverců v rámci sady. Jejich algortimus zkoumá všechny $2^{N-1} - 1$ oddíly N

bodů a vybere ten, který dává minimální součet čtverců.

3.2.1 Komentář k Metodě II

Výpočetní práce pro $2^{N-1}-1$ rozdělení je podle autorů enormní a nelze ji použít ani pro malé hodnoty N. Z toho důvodu se metoda nepoužívá. [4].

3.3 Metoda III (Williams a Lambert)

Metoda III zvažuje případ, kdy datovou matici X tvoří údaje o přítomnosti a nepřítomnosti označených jako 1 a 0. Jejich původní aplikace byly všechny ekologické, kde variace byly různé druhy rostlin přítomné (nebo nepřítomné) v N kvadrátech. N kvadráty mají být rozděleny do dvou podmnožin na základě druhu k, který je nejlépe odděluje. V jedné podmnožině kvadráty obsahují druh k a v druhé podmnožine jsou kvadráty bez druhu k. Vypočítá se matice R, jejíž prvky r_{ij}^2 jsou úměrné hodnotám x^2 odvozeným z kontingenční tabulky 2×2 druhů i versus druhů j. Druhy používané pro primární dělení jsou ty, pro které je

$$\sum_{i=1}^{v} x_{ik}^2 \tag{3.1}$$

hodnota maxima. k je druh, který má maximální součet řádků v R. Po primárním dělení se algoritmus opakuje na každé podmnožině, dokud některé pravidlo zastavení neukončí proces. Metoda poukazuje, že hodnota x_{ij}^2 je N násobkem čtverce korealce r_{ij} mezi druhy i a j. Autoři této metody uvádějí jako logiku, že pokud se dělí na základě variací x_k, pak součet čtvrců pro všechny ostatní jednotlivé variace x_i je dán obvyklým součtem čtverců vzhledem k regresnímu vzorci

$$r_{ik}^2 \sum_{i} (x_{ij} - \overline{x}_i)^2, \tag{3.2}$$

takže celkový součet všech proměnných lze napsat vztahem

$$\sum_{i=1}^{v} r_{ik}^2 \sum_{j=1}^{N} (x_{ij} - \bar{x}_i)^2. \tag{3.3}$$

3.3.1 Komentář k Metodě III

Výše uvedené zdůvodnění ukazuje, že stejně jako v Metodě II. je součet čtverců mezi oběma vytvořenými skupinami maximální, ale zde se zkoumá pouze rozdělení 2^{N-1} [4].

3.4 Srovnání Metod I, II a III

Metoda II vybírá z možných $2^{N-1}-1$ rozdělení, které maximalizuje součet čtverců. Metoda II dělá totéž, kromě toho, že se zkoumají pouze v možném rozdělení a dělení se provádí na jediném druhu (náhodné veličině), který maximalizuje součet čtverců. Avšak ani jedna z těchto metod není zcela uspokojivá, protože Metoda III může mít často nežádoucí vlastnosti a Metodu II lze využít pouze pro výpočet nízkých hodnot N.

Metoda I je oproti zbývajícím dvěma metodám metoda aglomerační, protože clustery se postupně rozšiřují přidáním dalších jedinců, nebo dříve definovaných clusterů. Metody II a III jsou metody dělící. V Metodě III je dělení prováděno na jediném znaku [4].

3.5 K-Means Metoda

Metoda K-Means pracuje na principu dělení. Funkce metody je taková, že dochází k rozdělování dat do vzájemně se vylučujících clusterů a vrací index clusteru, ke kterému přiřazuje pozorování. Narozdíl od hierarchického shlukování tato metoda pracuje na skutečných pozorováních a vytváří jedinou úroveň clusterů.

K-Means zachází s každým pozorováním v datech jako s objektem umístěným ve vesmíru. Nalezne oddíl, ve kterém jsou objekty v každém clusteru co nejblíže k sobě a co nejdále od objektů v jiných clusterech. Lze si vybrat z pěti různých měření vzdálenosti v závislosti na druhu dat, která jsou seskupována. Každý cluster v oddílu je definován členskými objekty a jeho centroidem, nebo centrem. Pro každý cluster definujeme centroid, což je bod, ke kterému je minimalizován součet vzdáleností od všech objektů v tomto clusteru. K-Means vypočítává centroidy clusterů odlišně pro každé měření vzdálenosti, aby minimalizovala součet s ohledem na míru, kterou určíme.

U této metody je použit iterativní algoritmus, který minimalizuje součet vzdáleností od každého objektu ke svému clusterovému centriodu přes všechny clustery. Tento algo-

ritmus přesouvá objekty mezi clustery, dokud nelze součet již dále snižovat. Výsledkem je sada clusterů, které jsou nejkopampaktnější a nejodolnější. Minimalizaci můžeme ovládat pomocí volitelných vstupních parametrů, včetně těch pro počáteční hodnoty centroidů clusteru a maximální počet iterací [3].

Syntaxe a popis

Syntaxe a popis platná pro program Matlab.

idx = kmeans(X,k) provádí dělení datové matice X typu $n \times p$ na k clustery a vrací vektor $n \times 1$ obsahující indexy clusteru každého pozorování, řádky odpovídají bodům a sloupce odpovídají proměnným.

idx = kmeans(X,k,Name,Value) vrací indexy clusteru s dalšími volbami určenými jedním, nebo více argumenty.

[idx,C] = kmeans() vrátí k clusterová centroidová místa v matici C typu $k \times p$.

[idx,C,sumd] = kmeans() vrací součet vzdáleností clusterů bod - centroid (těžiště).

[idx,C,sumd,D] = kmeans() vrátí vzdálenost od každého bodu ke každému centriodu v matici X typu $n \times k$.

Vstupní a výstupní argumenty

- X - data, vstupní argument
- k - počet clusterů, vstupní argument
- idx - index clusteru, výstupní argument
- C - umístění těžistě clusteru, výstupní argument
- sumd - součet clusterů vzdáleností od bodu k centroidu, výstupní argument
- D - vzdálenost od každého bodu ke každému těžišti, výstupní argument [3]

3.6 K-Medoids

K-Meloids clustering je metoda dělení, která se běžně používá v doménách, které vyžadují robustnost vůči odlehlým datům, metriky libovolné vzdálenosti, nebo pro které průměr a medián nemá jasnou definici. Princip je podobný jako u metody K-Means. Cílem obou metod je rozdělit sadu měření nebo pozorování do k podmnožin nebo clusterů tak, aby podmonižiny minimalizovaly součet vzdáleností mezi měřením a středem clusteru. V algoritmu K-means, střed podmnožiny je průměr měření v podmnožině, nazývá se centroid. V algoritmu K-Medoids je střed podmnožiny členem podmnožiny, nazýván medoid.

Algoritmus K-Medoids vrací medoidy, které jsou skutečnými datovými body v souboru dat. To umožňuje používat algortimus v situacích, kdy průměr dat v rámci sady dat neexistuje. Toto je hlavní rozdíl mezi K-Medoids a K-Means, kde centroidy vráceny pomocí K-Means nemohou být uvnitř souboru dat. Medoidy jsou tedy užitečné pro seskupování kategorických dat, kde průměr není možné definovat.

Funkce *kmedoids* poskytuje několik iterativních algoritmů, které minimalizují součet vzdáleností od každého clusteru medoid přes všechny clustery. Jeden z algoritmů se nazývá rozdělelní kolem mediodů (PAM), které probíhá ve dvou krocích [3].

1. Build-step: Každý z k clusterů je spojen s potenciálním medoidem. Toto přiřazení se provádí technikou určenou argumentem dvojice Name - Value „Start“.

2. Swap-step: V každém clusteru je každý bod testován jako potenciální medoid tím, že zkontroluje, zda se součet vzdáleností uvnitř clusteru zmenší pomocí tohoto bodu jako medoid. Pokud ano, bod je definován jako nový medoid. Každý bod je pak přiřazen clusteru s nejbližším medoidem.

Algoritmus opakuje kroky sestavení a výměny, dokud se medoidy nezmění, nebo jsou splněna jiná kritéria ukončení. Podrobnosti o minimalizaci lze ovládat pomocí několika volitelných vstupích pamametrů pro *kmedoids*, včetně těch pro počáteční hodnoty medoidů clusteru a pro maximální počet iterací [3].

Syntaxe a popis

Syntaxe a popis platná pro program Matlab.

idx = kmedoids(X,k) provádí K-Medoids clustering k rozdělení pozorování $n \times p$ matice X na k clustery a vrací $n \times 1$ vektor idx obsahující clusterové indexy každého pozorování, řádky X odpovídají bodům a sloupce odpovídají proměnným.

idx = kmedoids(X,k,Name,Value) používá další možnosti zadané jedním, nebo více argumenty Name, Value.

[idx,C] = kmedoids() vrací clusterová místa v clusteru v matici C typu $k \times p$.

[idx,C,sumd] = kmedoids() vrací součet vzdáleností mezi clustery bod-medoid vzdáleností ve vektoru $k \times 1$.

[idx,C,sumd,D] = kmedoids() vrací vzdálenosti od každého bodu ke každému medoidu v matici D typu $n \times k$.

[idx,C,sumd,D,midx] = kmedoids() vrací indexy, kde midx je vektor $k \times 1$.

[idx,C,sumd,D,midx,info] = kmedoids() vrací informace o struktuře s informacemi o možnostech použitých algoritmem při jeho spuštění.

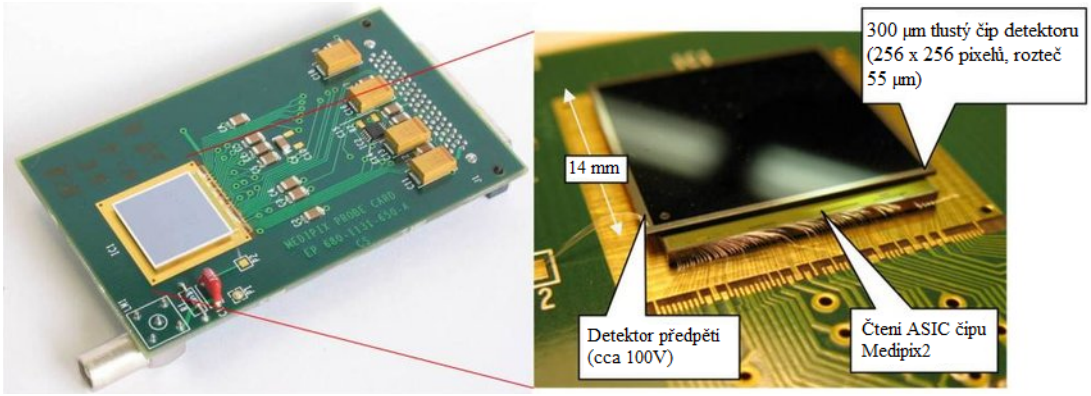
Vstupní a výstupní argumenty

- X - data, vstupní argument
- k - počet medoidů, vstupní argument
- idx - index medoidu, výstupní argument
- C - umístění clusterových medoidů, výstupní argument
- sumd - vnitřní součet vzdáleností clusterů od bodu k medoidům, výstupní argument
- D - vzdálenost od každého bodu ke každému medoidu, výstupní argument
- midx - index na x, výstupní argument
- info - informace o algoritmech, výstupní argument [3]

4 Pixelové detektory

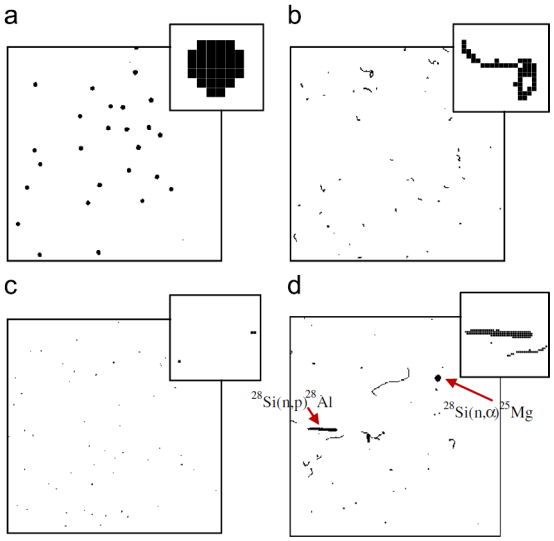
Lidské oko není natolik dokonalé, aby stačilo postřehnout letící částici. Právě proto byly vyvinuty pixelové detektory, které umožňují zachytit částice. Jsou různé typy částic a každá má svojí charakteristiku, která se promítne na senzorovém čipu [5].

4.1 Medipix2



Obrázek 4.1: Obrázek detektoru Medipix2, vpravo zvětšený pohled na čip [7].

Medipix2 je hybridní polovodičový pixelový detektor, který je tvořen ze senzorového čipu ($256 \times 256\,pixelů$, $každý\,55 \times 55\,\mu m$) a je spojen s výstupním čipem. Senzorový čip má citlivou plochu o rozměru přibližně $2cm^2$. V každém pixelu je obsažen předzesilovač, dva diskriminátory, 14-ti bitový pseudonáhodný čítač s detekcí přetečení a 8-bitový konfigurační registr. Konfigurační registr slouží pro maskování, testování a bitové nastavení prahových hodnot v jednotlivých pixelech. Takové pixelové zařízení může zaznamenávat jednotlivá kvanta ionizujícího záření. Pro různé typy záření je tato odezva odlišná. V případě, že čas závěrky je dostatečně krátký s ohledem na intenzitu záření, tak je možné jednotlivé stopy předmětů vidět a také je jasně od sebe oddělit. Track se skládá z pixelů, kde energie uložená nabitou částicí je nad prahovou hodnotou. Příklady různých záření můžeme vidět na Obr.4.2.



Obrázek 4.2: Reakce Medipix2 se senzorem o tloušťce 300 μm, když je osvětlen (a) zdrojem ^{21}Am alfa, (b) zdrojem ^{90}Sr beta, (c) zdrojem ^{21}Am XFR, (d) rychlými neutrony (až 30 MeV, 700 μm tlustý senzor) [5].

Můžeme vidět charakteristické kulaté tvary bloků částic, které pokrývají desítky pixelů v důsledku pronikání difuzí/sdílení mezi sousedními pixely (Obr.4.2.a). Vícenásobné rozptýlené elektrony vedou ke složeným stopám (Obr.4.2.b). Fotony 60 keV vysílají signály pouze v jediném pixelu, nebo v několika sousedních pixelech, typicky 1 - 4 (Obr.4.2.c). Protony vznikající z $^{28}Si(n,p)^{28}Al$ reakce v neutronovém svazku mohou generovat stub-like dráhy (Obr.4.2.d). Analýza těchto jednotlivých stop umožňuje stanovení složek neznámého pole záření. Takovým způsobem jsou stopy částic v křemíku v pevné fázi zobrazeny on-line podobným způsobem jako v jaderných emulzích, oblačných komorách nebo bublinových komorách [5].

4.2 Timepix Detector

4.2.1 Úvod

Zařízení Timepix je zcela nová generace pixelových detektorů CMOS. Timepix je navržená společností CERN a odvozená ze zařízení Medipix2. Umožňuje tři základní typy měření:

1. Time of Arrival (ToA) - čas výskytu

2. Time over Threshold (ToT) - doba nad prahovou hodnotou

3. Counting - počítání

V režimu Counting zvyšují pixely Timepix čítač pokaždé, když uložená energie překročí prahovou hodnotu. V režimu ToA zaznamenává pixel časovou značku záznamu a v režimu ToT udává hodnota pixelů celkovou uloženou energii. Tři provozní režimy podporované jedním čipem umožňují velký rozsah aplikací od astronomických pozorování, rentgenového fluorescenčního zobrazování až po rekonstrukci událostí ve fyzikálních experimentech (tj. analyzovat radioaktivní toky z neznámých radioaktivních zdrojů). Zařízení umožňuje vysokou vzorkovací frekvenci, může zaznamenávat až 3000 snímků za sekundu (fps-frame per second). Za snímačem je prahová hodnota umožňující potlačení šumu. To usnadňuje zpracování obrazu, protože ty pixely, které nedostávají dostatečné množství energie (nad prahovou hodnotou) z částice, obsahují nulovou hodnotu. V důsledku toho jsou získané obrázky považovány za bezšumové.

Princip rekonstrukce události je založen na rozpoznávání nabitých částic a jejich identifikační třídy následované některými statistickými měřeními (tj. hlavními částicemi a jejich úhlem dopadu). Rozpoznávání částic vyžaduje identifikovat a analyzovat stopu, která představuje „podpis“ částic. Různé částice zanechávají různě tvarované stopy v závislosti na typu částice, její energii a úhlu dopadu. Tudíž tvar a energie uložené podél každé dráhy může být použita pro identifikaci částic [6].

4.2.2 Klasifikace použitím morfologických charakteristik

Zařízení Timepix zaznamenává sekvenci šedých obrazů. Pracuje s rozměry rastru 256 × 256. Obrazy jsou skalární. V následujícím textu jeden cluster označuje připojenou komponentu nenulových obrazových bodů. Jeden cluster odpovídá stopě, kterou zanechala jedna částice (nebo více částic, pokud se překrývají). Ale v tomto případě předpokládáme, že jeden cluster je zanechán pouze jednou částicí. Každá připojená komponenta může být spojena s deskriptory umožňující klasifikaci částic do různých tříd. Nejčastěji se používají morfologické deskriptory (založené na analýze tvaru), které se zabývají plochou, promítanou a rozvinutou délkou, kostrou, kruhovitostí, atd. Klasifikační metody založené na deskriptorech jsou velmi účinné, bohužel jejich nevýhodou je výpočetní složitost. Vyžadují výpočet připojených komponent, označování, skeletonizaci a rekonstrukci a

to ještě před výpočtem deskriptorů. Tyto vlastnosti vycházejí z požadavků na vysokou paměť, nedefinované latence a pomalého výpočtu nepoužitelného v aplikacích s vysokým počtem snímků. Z těchto vlastností je zřejmé, že účinnost zpracování obrazu omezuje vzorkovací frekvenci snímání obrazu. Naproti tomu vysoká vzorkovací frekvence omezuje pravděpodobnost překrývajících se stop. Příspěvek nejprve prezentuje metodu rychlého třídění založenou na dvou deskriptorech a to na tloušťce a předpokládané délce. Tyto deskriptory se skládají z morfologické dilatace, eroze a jednoduchých aritmetických operátorů. Takový přístup se tak vyhne všem iterativním a nákladným algoritmům. Později navrhujeme modulární a parametrizovatelnou hardwarovou architekturu s využitím nedávno navržené rychlé implementace dilatačních a erozních operátorů. Vstupní obraz je klasifikován v jediném obrazovém skenování při velmi vysoké obnovovací frekvenci dosahující 738 fps při frekvenci 100 MHz. Navržená architektura je dále rozšiřitelná z hlediska počtu typů tras, které mají být klasifikovány [6].

4.3 Porovnání Medipix2 a Timepix detektorů

Oba tyto čipy pracují s maticí ($256 \times 256\,pixelů, každý\,55 \times 55\,\mu m$). Jak již bylo zmíněno výše, podobnost obou detektorů je vysoká, protože Timepix detektor je odvozen od detektoru Medipix2, ale funkce v rámci každého pixelu byla změněna. Hlavní rozdíl je v možnostech měření. V detektoru Timepix může být každý pixel naprogramován tak, aby počítal zásahy stejně jako Medipix2, ale navíc umožňuje dvě další měření. Jedná se o měření ToA (Time of Arrival) a ToT (Time over Threshold) [10].

4.4 Klasifikace tracků

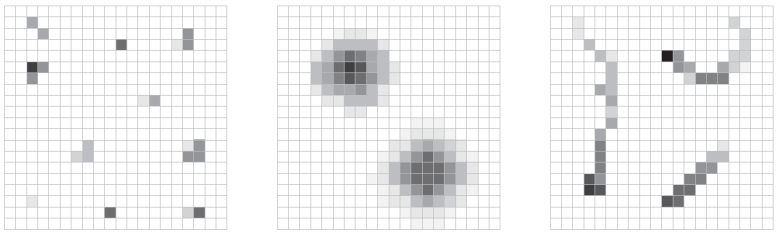
Následující algoritmy byly implementovány do softwarového balíčku Pixelman jako plugin pro online analýzu clusterů. Zaznamenaný vzor interakce je analyzován a rozdělen na jednotlivé stopy. To se provádí osmicestnou vyhledávací rutinou, která najde „nepřetržité" seskupení. V ideálním případě, když se žádné stopy nepřekrývají, by měl být registrovaný vzorek binárním obrazem se zónami, kde registrovaný náboj byl nad vybranou prahovou hodnotou a nulami jinde. Překrývající se stopy proto mohou být snadno detekovány a vyřazeny. Je možné se pokusit oddělit dvě překrývající se stopy. Obecně to však

není tak snadné a v současnosti se to neprovádí. Takže pokud je počet překrývajících se stop významný, musí být doba expozice zkrácena, aby se snížila pravděpodobnost těchto událostí. Klasifikace do předdefinovaných kategorií je založena na geometrických prvcích popisující clustery.

V této studii se zabývám třemi hlavními třídami stop, které se nazývají:

- blobs - kuličky
- dots - tečky
- tracks - stopy

Tyto názvy odpovídají terminologii jaderné fyziky. Dots jsou generovány například nízkoenergetickými elektrony nebo fotony. Blobs jsou zanechány těžkými ionty. Lineární, nebo vlnité stopy jsou produkovány minimálními ionizujícími částicemi, nebo elektrony.

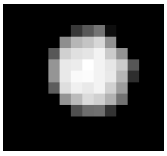


(a) dots (b) blobs (c) tracks

Obrázek 4.3: Příklady stop uložených různými částicemi [6].

Dále je možno hlavní tři třídy rozdělit do dalších podmnožin, které se nazývají:

- dot - energie fotonu < 20 keV
- small blob - energie fotonu ~ 50 keV
- heavy blob - alfa částice, těžké ionty, nebo pomalé neutrony [6]

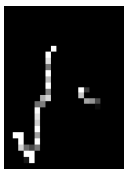


Obrázek 4.4: Stopa alfa částice [8].

- straight thin track - minimální ionizující částice gama

Obrázek 4.5: Stopa gama částice [8].

- curly track - elektrony, nebo elektrony produkované fotony $>$ 50 keV, lze také nazývat beta částice



Obrázek 4.6: Stopa beta částice [8].

- heavy track - protony $>$ 1MeV, neutrony $>$ 1 MeV

Obrázek 4.7: Stopa částice protonu [8].

Stejná částice může vytvářet clustery různého typu (nebo s různými parametry) při různých nastaveních prahu a zkreslení. Příklady pro jednotlivé kategorie byly zvažovány pro poměrně nízké prahové nastavení. Pokud je například prahová hodnota zvýšena, alfa částice se zmenší, protože náboj shromážděný v sousedním pixelu nepřekročí prahovou hodnotu.

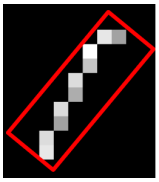
Geometrické rysy, které jsou extrahovány pro každý cluster, zahrnují konvexní trup, plochu, objem, počet vnitřních/hraničních pixelů, délku hranic, maximální počet pixelů na přímce, maximální vzdálenost v clusterech, atd. Funkce jsou používány pro výpočet parametrů, které definují parametrický prostor pro klasifikaci. Pokud parametry určitého clusteru spadají do určitých částí parametrického prostoru, je cluster přiřazen odpovídající kategorii. Například poměr průměru vypočteného z oblasti clusteru a maximální vzdálenosti

v clusteru by měl být roven jedné pro dokonale kruhové clustery. Pokud takto vypočítaný parametr spadá do určitého intervalu kolem jedné, je cluster dostatečně kulatý a pokud jsou splněny jiné podmínky, jako je minimální počet vniřních pixelů, je cluster klasifikován jako „heavy blob". Tyto podmínky, které rozdělují parametrický prostor do kategorií lze upravit s ohledem na podmínky daného experimentu a nastavení zařízení. Je-li stopa odpovídající jedné částici nespojitá, může se algoritmus pokusit spojit tyto clustery dohromady. To se provádí konvolucí obrazu tvořeného „non-heavy" clustery (které jsou pravděpodobně přerušované) s dvourozměrným Gaussovým jádrem a spojí se, pokud mezi nimi existuje cesta nad zvolenou prahovou hodnotou. Tímto způsobem jsou clustery spojeny ve směru existujících stop [6].

Výstup ze zařízení Timepix lze tedy popsat funkcí $I : t \times (x, y) \rightarrow E; (x, y) \in Z^2, E \in Z \cap [0, 65535]$, kde x a y jsou souřadnice, E je energie uložená a $t \in R^2$ je čas pořízení snímku. Jak je uvedeno v kapitole 4.2.2 se zabýváme případem, kdy se stopy částic nemohou překrývat a cluster odpovídá právě jedné stopě. Každý cluster můžeme popsat clusterovými charakteristikami. Cluster lze definovat jako vektor $f(c_i) = V; f : C \rightarrow S; S \subset R^N$, kde N je počet použitých clusterových charakteristik, f je funkce, která vypočítává clusterové charakteristiky a S je prostor, který obsahuje všechny N-rozměrné vektory charakteristik, které odpovídají různým clusterům. Cílem tohoto je rozdělit clustery do patřičných kategorií. Každá kategorie odpovídá právě jednomu typu částice.

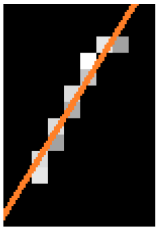
Jednou z clusterových charakteristik může být například charakteristika linearity, jejíž hodnoty nabývají od 0 do 1. V případě, že se linearita blíží k jedné, znamená to, že se jedná o přímku. Naopak, když se linearita blíží k nule, tak se jedná o nelinearitu. Bohužel nelinearita záleží na matematické definici a proto používáme dvě definice. První je poměr šířky k výšce ohraničujícího obdélníku, který obsahuje cluster. Tento poměr je vyjádřen vztahem 4.1 a grafické znázornění je vidět na Obr. 4.8 [8].

$$lin = 1 - \frac{min(width, height)}{max(width, height)} \tag{4.1}$$



Obrázek 4.8: Linearita [8].

Druhou definici lze vysvětlit tak, že se jedná o kvadratickou odchylku od lineární funkce. Odchylka je normalizována na [0,1], kde 1 odpovídá tvaru kruhu (Obr. 4.9).



Obrázek 4.9: Lineární fit [8].

5 Funkce pro hledání clusterů v programu Matlab

V této kapitole jsou vybrány a popsány některé funkce, které je vhodné použít pro hledání clusterů v Matlabu. U každé funkce je uvedena syntaxe a popis, které jsou převzaty z [3].

5.1 pdist

Hledáme podobnost a nepodobnost mezi párovými objekty v matici dat, kde zjišťujeme vzdálenost mezi jednotlivými objekty za pomocí funkce **pdist**. Výstupem funkce je matice vzdáleností, nebo nepodobností [9].

Syntaxe

D = pdist(X)

D = pdist(X,distance)

Popis

D = pdist(X) umožňuje určit euklidovskou vzdálenost mezi dvojicemi objektů a to v matici dat X o rozměrech $m \times n$, kde řádky odpovídají pozorováním a sloupce proměnným. Výsledkem je řádkový vektor D o délce $m(m-1)/2$, který odpovídá dvojicím pozorování v X. Uspořádání vzdáleností je pak následující (2, 1), (3, 1), ...,(m, 1), (3, 2), ..., (m, 2), ..., (m, m-1). Výsledný vektor D lze však pomocí funkce squareform zapsat jako matici.

D = pdist(X,distance) umožňuje dopočítat mezi objekty v zadané datové matici X jejich vzdálenost a to pomocí vybrané metody metriky z Tab. 5.1:

Tabulka 5.1: Možnosti metrik pro výpočet vzdálenosti objektů [3].

Metrika	**Popis**
'euclidean'	Euklidovská vzdálenost.
'seuclidean'	Standardizovaná Euklidovská vzdálenost.
'cityblock'	Metrika městského bloku.
'minkowski'	Minkowskiho vzdálenost, která používá výchozí exponent číslo 2.
'chebychev'	Chebychevova vzdálenost.
'mahalanobis'	Mahalanobisova vzdálenost.

5.2 linkage

Funkce **linkage** nám umožňuje spojit páry objektů, které jsou si podobné do hierarchického stromu [9].

Syntaxe

Z = linkage(X)

Z = linkage(X,method)

Z = linkage(X,method,metric)

Z = linkage(X,method,pdistinputs)

Z = linkage(X,method,metric,'savememory',value)

Popis

Z = linkage(X) vrátí matici, která z reálné matice X kóduje hierarchický strom clusterů.

Z = linkage(X,method) vytvoří strom pomocí metody, kterou lze libovolně vybrat. Vybraná metoda popisuje, jak se bude měřit vzdálenost mezi clustery.

Z = linkage(X,method,metric) provede clustering za účelem výpočtu vzdáleností mezi řádky x a to pomocí metriky meření vzdáleností.

Z = linkage(X,method,pdistinputs) předá parametry funkci pdist, která umožní spočítat vzdálenost mezi řádky x.

Z = linkage(X,method,metric,'savememory',value) používá algoritmus pro ukládání paměti v případě, že value (hodnota) je ve stavu true. Když je value ve stavu false, použije se standardní algoritmus.

5.3 cluster

Funkci cluster se používá v případěch, kdy je potřeba změnšit větve v blízkosti spodní části heirarchického stromu [9].

Syntaxe

T = cluster(Z,'cutoff',c)

T = cluster(Z,'cutoff',c,'depth',d)

T = cluster(Z,'cutoff',c,'criterion',criterion)

T = cluster(Z,'maxclust',n)

Popis

T = cluster(Z,'cutoff',c) konstruuje clustery z aglomeračního hierarchického clusterového stromu. Z je matice o velikosti $(m-1)$, kde m je počet pozorování v původních datech. Písmeno c vyjadřuje prahovou hodnotu pro řez Z do clusterů. Clustery jsou tvořeny, když uzel a všechny jeho poduzly mají nekonzistentní hodnotu menší než je velikost prahové hodnoty c.

T = cluster(Z,'cutoff',c,'depth',d) vyhodnocuje nekonzistentní hodnoty pohledem do hloubky d pod každým uzlem. Výchozí hodnota hloubky je 2.

T = cluster(Z,'cutoff',c,'criterion',criterion) používá zadané kritérium pro vytváření clusterů, kde kritériem je jeden z nekonzistentních řetězců, nebo vzdálenost. Kritérium vzdálenosti používá vzdálenost mezi dvěma poduzly sloučenými v uzlu

pro měření výšky uzlu. Všechny listy v uzlu, nebo pod ním s výškou menší než c jsou seskupeny do clusteru.

T = cluster(Z,'maxclust',n) konstruuje maximálně n clusterů pomocí kritéria vzdálenosti. Cluster najde nejmenší výšku, při které vodorovný řez stromem ponechá n, nebo méně clusterů.

5.4 clusterdata

Syntaxe

T = clusterdata(X,cutoff)

T = clusterdata(X,Name,Value)

Popis

T = clusterdata(X,cutoff) vrátí indexy clusteru T pro každé pozorování dat X při dodržení prahu pro řez hierarchického stromu (cutoff).

T = clusterdata(X,Name,Value) clustery s dalšími volbami určenými jedním nebo více argumenty.

Vstupní argumenty

- X - matice se dvěma, nebo více řádky, řádky představují pozorování, sloupce představují kategorie nebo rozměry
- cutoff - když $0 < cutoff < 2$, clusterdata vytváří clustery, když nekonzistentní hodnoty jsou větší než cutoff a když cutoff je celé číslo ≥ 2, clusterdata interpretuje cutoff jako maximální počet clusterů, které mají být udržovány v hierarchickém stromu generovaném vazbou

Argumenty dvojice Name-Value

Zadává se dvojice argumentů (lze zadat i několik za sebou v libovolném pořadí) Name a Value, které se od sebe oddělí čárkami. Name je název argumentu a Value je odpovídající hodnota. Name se zapisuje v uvozovkách (' ').

- 'criterion' - argument vyjadřující nekonzistentnost, nebo vzdálenost
- 'depth' - hloubka pro výpočet nekonzistentních hodnot zadavána jako kladné celé číslo
- 'distance' - libovolný název metrických vzdáleností
- 'linkage' - libovolná metoda propojení
- 'maxclust' - maximální počet clusterů, které se mají vytvořit, kladné celé číslo
- 'savememory' - argument mající dva stavy „on" nebo „off"

Výstupní argumenty

- T - vektor velikosti m obsahující číslo clusteru pro každé pozorování

Když $0 < cutoff < 2$, T = clusterdata(X,cutoff) je ekvivalentní:

Y = pdist(X,'euclid');
Z = linkage(Y,'single');
T = cluster(Z,'cutoff',cutoff);

Když cutoff je celé číslo ≥ 2, T = clusterdata(X,cutoff) je ekvivalentní:

Y = pdist(X,'euclid');
Z = linkage(Y,'single');
T = cluster(Z,'maxclust',cutoff);

5.5 regionprops

Syntaxe

stats = regionprops(BW,properties)

stats = regionprops(CC,properties)

stats = regionprops(L,properties)

stats = regionprops(...,I,properties)

stats = regionprops(output,...)

stats = regionprops(gpuarrayImg,...)

Popis

stats = regionprops(BW,properties) vrací měření pro sadu vlastností určených vlastnostmi pro každou připojenou komponentu (objekt) v binárním obrazu BW.

stats = regionprops(CC,properties) vrací meření pro sadu vlastností určených vlastnostmi pro každou připojenou komponentu (objekt) v CC, kde CC je vrácená struktura.

stats = regionprops(L,properties) vrací meření pro sadu vlastností specifikovaných vlastnostmi pro každou označenou oblast v matici L.

stats = regionprops(...,I,properties) vrací meření pro sadu vlastností určených vlastnostmi pro každou označenou oblast v obraze I.

stats = regionprops(output,...) vrací měření pro sadu vlastností, kde výstup určuje typ návratové hodnoty, funkce regionprops může tyto hodnoty vrátit do struktury, nebo do tabulky.

stats = regionprops(gpuarrayImg,...) provádí meření na GPU, kde gpuarrayImg může být 2D binární obraz, nebo matice.

Vstupní argumenty

- BW - vstupní binární obraz
- properties - typ měření
- CC - připojené komponenty
- L - označené oblasti
- I - obrázek, který má být měřen
- output - návratový typ
- gpuarrayImg - vstupní 2D obraz, nebo matice

Typ měření pro argument *properties*

Tabulka 5.2: Argumenty pro měření tvaru [3].

Název vlastnosti	**Popis**
'Area'	Vrací skalár, který skutečný počet pixelů v oblasti.
'Centroid'	Vrátí vektor $1 \times Q$, který specifikuje centroid (težiště) oblasti. Prvním prvkem centroidu je horizontální souřadnice (také lze označit jako souřadnice x) a druhým prvkem je svislá souřadnice (lze označit jako souřadnice y).
'Image'	Vrátí binární obraz stejné velikosti jako ohraničovací rámeček oblasti. Pixely odpovídají oblasti a všechny ostatní pixely jsou vypnuty.
'PixelList'	Vrátí matici $p \times Q$, která určuje umístění pixelů v oblasti. Každý řádek matice má tvar $[x\, y\, z \ldots]$ a určuje souřadnice jednoho pixelu v oblasti.
'Perimeter'	Vrátí skalár, který určuje vzdálenost kolem hranice oblasti. regionprops vypočítá obvod vzdálenosti mezi každým sousedním párem pixelů kolem hranice oblasti. Pokud obraz obsahuje nesouvislé oblasti, tak regionprops vrátí špatné výsledky.

Tabulka 5.3: Argumenty pro měření hodnoty pixelů [3].

Název vlastnosti	**Popis**
'MaxIntensity'	Vrací skalár, který určuje hodnotu pixelu s největší intenzitou v oblasti.
'MeanIntensity'	Vrací skalár, který udává průměr všech hodnot intenzity v oblasti.
'MinIntensity'	Vrací skalár, který určuje hodnotu pixelu s nejnižší intenzitou v oblasti.
'PixelValues'	Vrací vektor $p \times 1$, kde p je počet pixelů v oblasti. Každý prvek obsahuje hodnotu pixelu v oblasti.
'WeightedCentroid'	Vrací vektor $p \times Q$ souřadnic určující střed oblasti na základě polohy a hodnoty intenzity. Prvním prvkem je vodorovná souřadnice (lze označit souřadnice x) váženého těžiště. Druhým prvkem je svislá souřadnice (lze označit souřadice y).

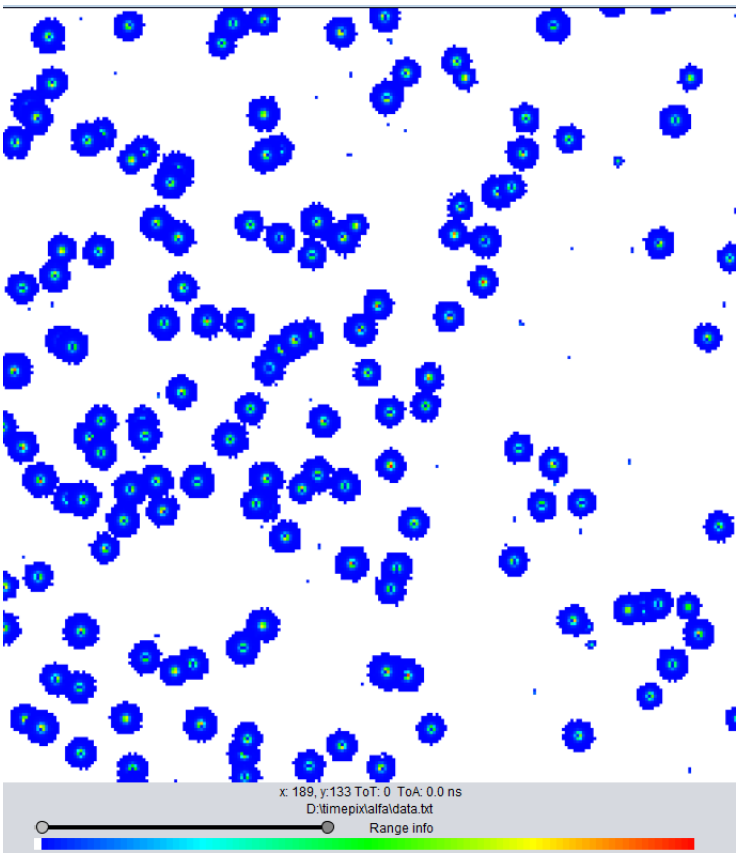
Výstupní argumenty

- stats - hodnoty měření

6 Řešení zadané matice v programu Matlab

Naměřená data alfa částic z detektoru Timepix byla zadána v textovém souboru, který obsahoval čtyři sloupce o 13 781 řádcích a zobrazen na Obr. 6.1. Sloupce byly v pořadí:

1. $coor = 256 \cdot x + y$
2. $toaOverall$
3. $fineToa$
4. $tot - Time\,over\,Threshold$



Obrázek 6.1: Naměřená data alfa částic z Timepix detektoru [11].

Původní textový soubor jsem z důvodu překrývání clusterů zredukoval na velikost 1 186 řádků a počet sloupců zůstal stejný (4). Textový soubor jsem načetl do Matlabu. Rozdělil jsem ho na jednotlivé sloupcové vektory pro lepší práci v dalším používání. Definoval jsem

dále vektory TimeoverThreshold a TimeOfArrival. Vektor TimeOfArrival jsem dopočítal podle vztahu

$$TimeOfArrival = toaOverall \cdot 25 + fineToa \cdot 1,562\,[\text{ns}]\,. \tag{6.1}$$

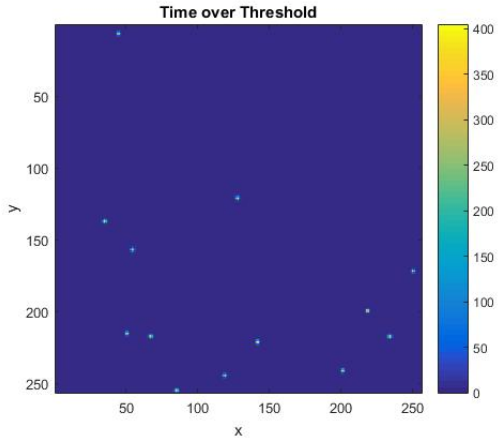
Následně jsem zjistil nejnižší a nejvyšší hodnotu v TimeOfArrival a poté za pomoci for cyklu získal matice tot a toa.

```
%načtení naměřených dat jako matice o čtyřech sloupcích
Y = dlmread('podklady2pul.txt')

%odseparování matice na jednotlivé sloupcové vektory
coor = Y(1:length(Y),1);
toaOverall = Y(1:length(Y),2);
fineToa = Y(1:length(Y),3);
totinit = Y(1:length(Y),4);

%Time over Threshold
TimeoverThreshold = zeros(length(Y),1); %vektor nul o délce
%vektoru Y
TimeoverThreshold(1:length(Y),1) = totinit(1:length(Y),1);
%TimeoverThreshold = totinit

%Time of Arrival
TimeOfArrival= zeros(length(Y),1); %vektor nul o délce vektoru Y
TimeOfArrival(1:length(Y),1) = toaOverall(1:length(Y),1)*25 +
+fineToa(1:length(Y),1)*1.562 %[ns] %výpočet TimeOfArrival
%podle vzorce

%určení minima a maxima v TimeOfArrival
```

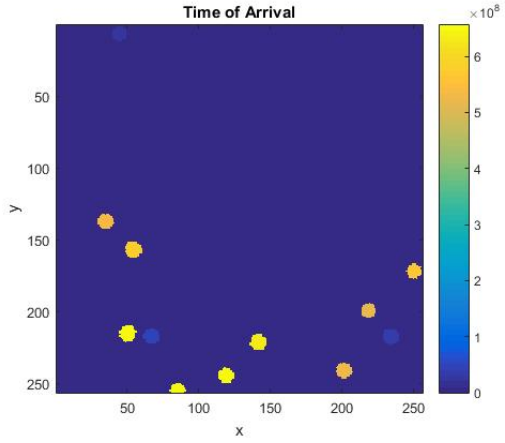
```
low=min(TimeOfArrival); %výpočet minima TimeOfArrival
high=max(TimeOfArrival); %výpočet maxima TimeOfArrival

toa = zeros(256,256); %deklarace matice nul Time to Arrival
tot = zeros(256,256); %deklarace matice nul Time over Threshold

[r,s] = size(Y); %uložení počtu řádků r a počtu sloupců s matice Y

for x = 1:256 %určuje pozici řádku v matici 256x256
        for y = 1:256 %určuje pozici sloupce v matici 256x256
                val = 256*(x-1)+(y-1); %výpočet hodnoty souřadnice
                for t = 1:r %proměnná t prochází coor
                                        %(první sloupec matice Y)
                        if(coor(t)== val) %v případě shody
        %souřadnic se zapíše na místo určené x,y
                                toa(x,y) = toaOverall(t)*25 +
        +fineToa(t)*1.562; %výpočet zapsání Time of Arrival
                                        %do matice toa
                                tot(x,y) = TimeoverThreshold(t) ;
        %zapsání Time over Threshold do matice tot
                        end
                end
        end
end

close all %zavření všechn oken
```

Některé funkce vyžadují práci pouze s binárními čísly. Proto jsem v dalším kroku převedl matici tot na binární (totBW) a všechny tři matice (tot, toa, totBW) vykreslil.

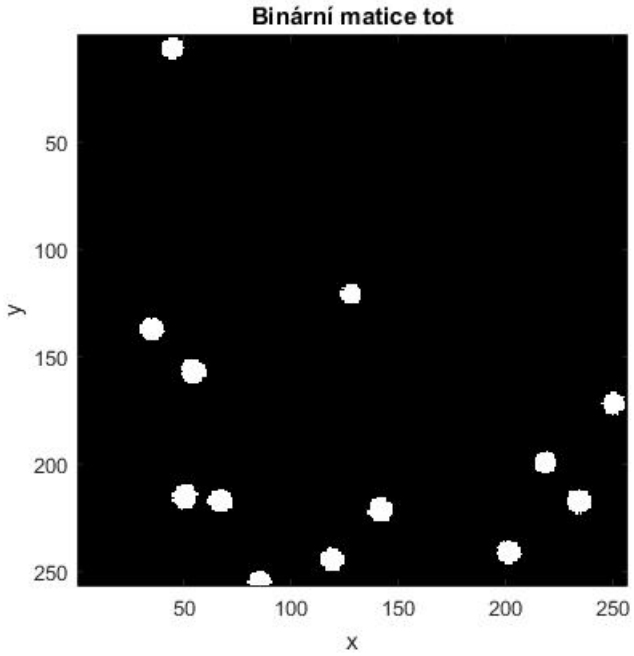
```
image(tot,'CDataMapping','scaled') %vykreslení tot
title('Time over Threshold') %název obrázku
xlabel('x') %název osy x
ylabel('y') %název osy y
axis square; %čtvercový tvar
colorbar %barevná osa
figure()

image(toa,'CDataMapping','scaled') %vykreslení toa
title('Time of Arrival')
xlabel('x')
ylabel('y')
axis square;
colorbar
figure()

totBW = logical(tot) %tot převedena na binární matici
image(totBW,'CDataMapping','scaled')
title('Binární matice tot')
xlabel('x')
ylabel('y')
axis square;
colormap bone
figure()
```



Obrázek 6.2: Vykreslení matice Time over Threshold (tot).



Obrázek 6.3: Vykreslení matice Time of Arrival (toa).

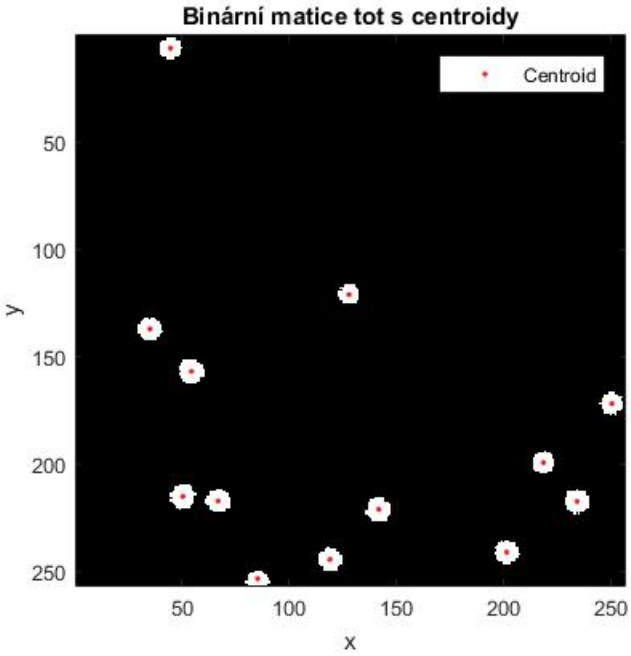


Obrázek 6.4: Vykreslení binární matice tot (totBW).

Z matice totBW jsem určil těžiště clusterů.

```
%určení těžiště clusterů
stats = regionprops(totBW,'Centroid');
centroids = cat(1,stats.Centroid); %cat - spojí matice podle
```

```
%zadaného rozměru

%vykreslení binární matice s centroidy
image(totBW,'CDataMapping','scaled') %vykreslení binární matice tot
title('Binární matice tot s centroidy') %název obrázku
xlabel('x') %název osy x
ylabel('y') %název osy y
axis square;
colormap bone
hold on
plot(centroids(:,1),centroids(:,2),'r.') %vyznačení centroidů
legend('Centroid') %legenda
hold off
```



Obrázek 6.5: Vykreslení binární matice s centroidy.

Po zjištění centroidů clusterů jsem dále určil počet clusterů v obrázku, obvod clusterů a vzdálenost mezi nimy. U zjišťování obvodů clusterů jsem provedl meření rychlosti zpra-

cování funkce regionprops. Dále bylo zjištěno, kolik pixelů každý cluster obsahuje a jak jsou pixely umístěny. Všechny tyto zjištěné vlastnosti jsem vypsal do textového souboru s názvem Výsledky.txt.

```
pocet = length(stats); %počet clusterů v obrázku (odpovídá počtu
%centroidů)

%měření rychlosti zpracování výpočtu obvodů
casvysledny = 0; %vytvoření proměnné pro výsledný čas
for stokrat = 1:100 %for cyklus pro měření času zpracování (stokrát)
        tic
        p = regionprops(totBW,'Perimeter'); %zjištění obvodu
        %clusterů
        casvysledny = casvysledny + toc; %celkový čas
        %po stonásobném spuštění
end
cas = casvysledny/100 %výpočet času pro zjištění rychlosti určení
%obvodu clusterů

p_typ = struct2array(p); %přetypování p na double

measurements = regionprops(totBW, 'Area'); %počet pixelů v
%clusterech
measurements_typ = struct2array(measurements); %přetypování
%measurements na double

location_pix = regionprops(totBW, 'PixelList'); %matice umístění
%pixelů
```

```
%zápis do textového souboru
f = fopen('Výsledky.txt','w') %otevření txt souboru s výsledky

fprintf(f, 'Počet clusterů: %d\n', pocet); %celkový počet clusterů
disp('Počet clusterů je: ') %vypsaní počtu clusterů na obrazovku
disp(pocet)

fprintf(f, 'Celková doba zpracování funkce regionprops: %.4f[s]\n',
cas); %doba zpracování
fprintf(f, 'Index  Počet pixelů v clusteru: \n');
for prom = 1:pocet
        fprintf(f, '%d.\t\t %d\n', prom, measurements_typ(prom));
end
disp('Počet pixelů v clusteru je: ') %vypsání počtu pixelů v
%clusteru na obrazovku
disp(measurements_typ)

fprintf(f, 'Index  Obvody clusterů: \n'); %výpis obvodů clusterů
%do souboru
for prom = 1:pocet
        fprintf(f, '%d.\t\t %d\n', prom, p_typ(prom));
end
disp('Obvody clusterů jsou: ') %výpis obvodů clusterů na obrazovku
disp(p_typ)
```

7 Závěr

Bakalářská práce se zabývala tím, co je to pojem Cluster analýza, k čemu se používá a jaké metody nabízí. Práce se také věnovala pixelovým detektorům a krátce ukázala jejich funkci a to, jak vůbec v reálné podobě vypadají. V druhé části práce byl předveden algoritmus, který umí řešit naměřená data právě z pixelového detektoru.

Čtyřem kapitolám byla přidělena rešerše. Druhá kapitola pojednává o tom, co to samotná Cluster analýza je a v jakých aplikacích se dá využít. V třetí kapitole byly zmíněny metody, které se využívají v této problematice. První tři metody jsou historické a zbylé dvě se v současnosti hojně využívají. Následující kapitola konkrétně popsala a ukázala dva pixelové detektory. První detektor Medipix2 si bylo možné představit z obrázku, který byl uveden hned za začátku této kapitoly. Také bylo řečeno, jaké možnosti měření se na něm dají využít. Druhý detektor, Timepix, je nástupcem předchozího detektoru a jeho konstrukce je obdobná. Výhody této metody spočívají zejména v možnostech typů měření a jsou opět zmíněny hned v úvodu popisovaného detektoru. Po představení obou detektorů jsem využil srovnání, ze kterého lépe vyšel druhý detektor Timepix. Pátá kapitola se zaměřuje na funkce, které by se mohly použít pro hledání vlastností clusterů v programu Matlab. U každé funkce byla uvedena syntaxe i popis, aby bylo ukázáno, jak lze dotyčnou funkci zapsat. Některé funkce byly použity v následujícím algoritmu.

Závěr práce byl vymezen pro vlastní algoritmus v programu Matlab. Ze zadaných dat, se po převedení do matic dal určit například počet clusterů, medoidy (těžiště) jednotlivých clusterů a počet pixelů v jednotlivých clusterech. Byla zjištěna i doba měření rychlosti funkce regionprops. Doba změřená pro tato data byla 0,0034s. Dle mého názoru je to velmi krátká doba zpracování. Tohoto algoritmu by se dalo využít i pro online řešení, ale pouze v případě, že by se clustery nepřekrývaly. Pokud se clustery překrývají, dojde ke špatnému určení medoidů a následně ke špatným výpočtům v algoritmu.

Reference

[1] GOWER, J. C. A Comparison of Some Methods of Cluster Analysis. Biometrics. 1967, 23(4). DOI: 10.2307/2528417. ISSN 0006341X. Dostupné také z: https://www.jstor.org/stable/2528417?origin=crossref

[2] What is cluster analysis [online]. 2016 [cit. 2019-05-09]. Dostupné z: https://www.slideshare.net/Prabhatgangwar/what-is-cluster-analysis

[3] The MathWorks [online]. United Kingdom, 1994 [cit. 2019-05-09]. Dostupné z: https://uk.mathworks.com/

[4] REICHL, Jaroslav a Martin VŠETIČKA. Encyklopedie fyziky [online]. 2006 [cit. 2019-06-09]. Dostupné z: http://fyzika.jreichl.com/main.article/view/846-detektory-castic

[5] HOLY, T., E. HEIJNE, J. JAKUBEK, S. POSPISIL, J. UHER a Z. VYKYDAL. Pattern recognition of tracks induced by individual quanta of ionizing radiation in Medipix2 silicon detector. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment. 2008, 591(1), 287-290. DOI: 10.1016/j.nima.2008.03.074. ISSN 01689002. Dostupné také z: https://linkinghub.elsevier.com/retrieve/pii/S0168900208004592

[6] BARTOVSKÝ, J., SCHNEIDER, D., DOKLADALOVA, E., DOKLADAL, P., GEORGIEV, V., AKIL, M. Morphological Classification of Particles Recorded by the Timepix Detector. In Proceedings of the 7th IS on Image and Signal Processing and Analysis. Zagreb, Croatia: University of Zagreb, 2011. s. 343-348. ISBN: 978-953-184-159-7 , ISSN: 1845-5921

[7] Using MARS Spectral CT for Identifying Biomedical Nanoparticles - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Left-A-photo-of-Medipix2-detector-Right-Magnified-view-of-detector-from-left-Top-and_fig8_257029228 [accessed 9 May, 2019]

[8] ČERMÁK, Jakub. Rozpoznávání drah částic v pixelovém detektoru typu Timepix. Praha, 2013. Diplomová práce. Univerzita Karlova v Praze.

[9] GREGOROVÁ, Viera. Zhluková analýza v systémoch STATISTICA a MATLAB. Brno, 2008. Diplomová práce. Masarykova univerzita v Brně.

[10] Medipix [online]. Ženeva: CERN [cit. 2019-05-20]. Dostupné z: https://medipix.web.cern.ch/

[11] J.Broulím, P.Broulím, P.Burian, M.Holík, Y.Mora, S.Pospíšil, M.Solar, "j-Pix - A multiplatform acquisition package for Timepix3". Journal of Instrumentation, 2019.

Přílohy

Všechny přílohy jsou dostupné v elektronické podobě na přiloženém CD.