Rozpoznawanie statycznych gestów wykonywanych dłonią na podstawie danych 3D

Wykład w ramach przedmiotu „Metody rozpoznawania obiektów i analizy ruchu”

Automatyczne rozpoznawanie układów dłoni - motywacja

- Zastosowania automatycznego rozpoznawania gestów:
  1. Interpretacja wypowiedzi w języku migowym
  2. Interakcja człowiek-robot
  3. Sterowanie urządzeniami w inteligentnym mieszkaniu
  4. Operowanie wirtualnymi obiektami

Etapy rozpoznawania gestów na podstawie deskryptorów chmur punktów

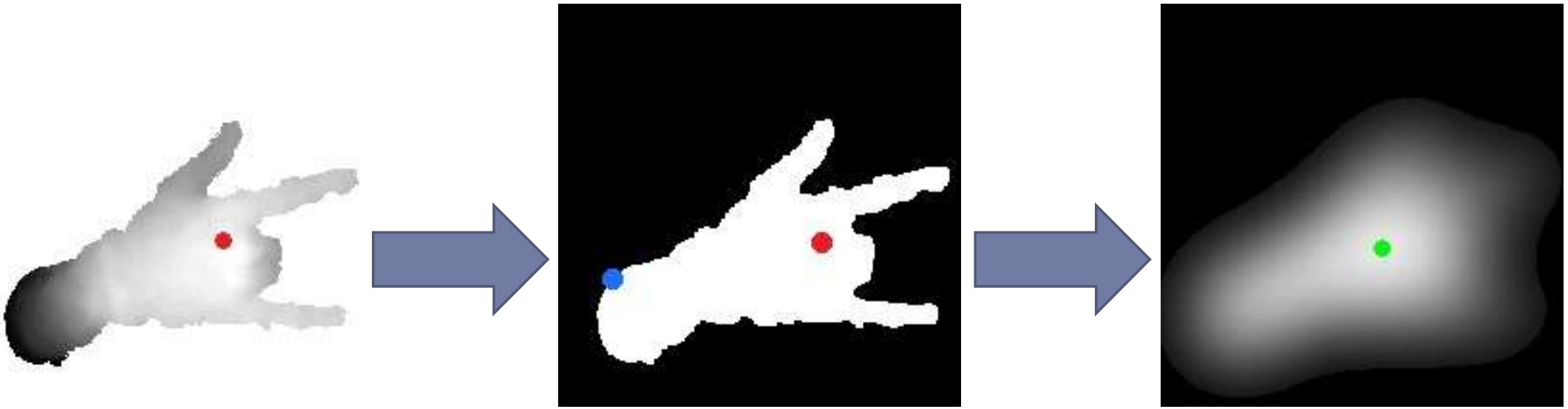
1. Pozyskiwanie danych w formie map głębi z sensora (kamery)
2. Segmentacja dłoni (oddzielenie pikseli dłoni od tła oraz innych części ciała)
3. Obrócenie dłoni do pionu [opcjonalne]
4. Konwersja mapy głębi na chmurę punktów*
5. Filtracja i downsampling chmury punktów [opcjonalne]
6. Wyznaczenie cech deskryptorów
7. Normalizacja cech
8. Uczenie klasyfikatora (na podstawie danych treningowych)
9. Klasyfikacja

* Jeśli segmentację wykonujemy na chmurach punktów, a nie mapach głębi, konwersję należy przeprowadzić po kroku 1.

Przykładowa metoda segmentacji dłoni - rozpoznawany gest i ograniczenia metody

Wymaganie: Dłoń powinna być obiektem najbliższym od kamery, oddalonym o co najmniej 10 cm od tułowia.

Przykładowa metoda segmentacji dłoni – cz. I

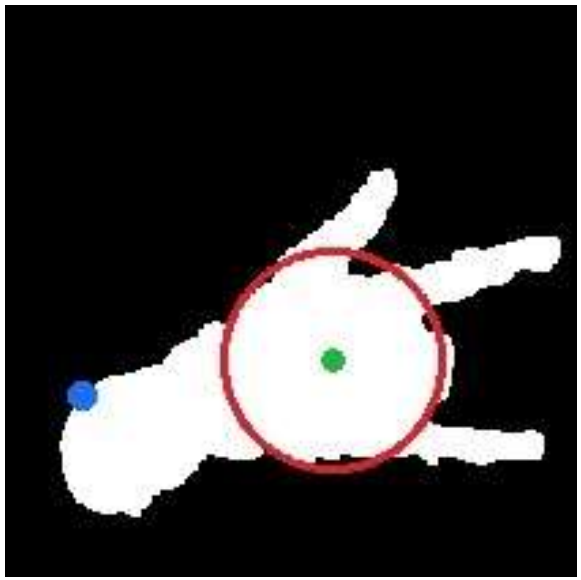


- Odcięcie pikseli o głębi większej niż 10 cm od punktu najbliższego od kamery *N* (czerwona kropka).

- Binaryzacja mapy głębi
- Wyznaczenie punktu najdalszego od kamery *F* (niebieska kropka).

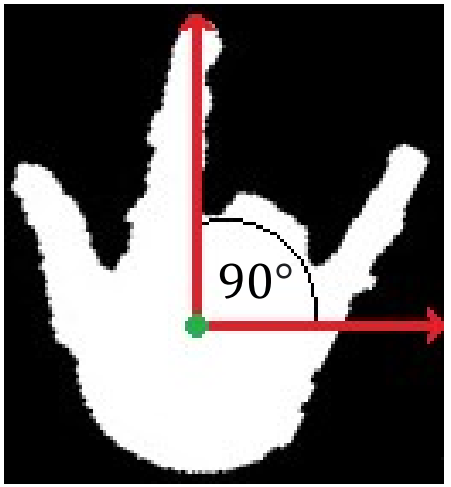
- Filtracja Gaussa (z dużą maską przykrywającą całą dłoń).
- Wyznaczenie punktu środkowego dłoni C (zielona kropka) będącego maksimum lokalnym, które nie jest zbyt bardzo oddalone od punktu *N.*

Przykładowa metoda segmentacji dłoni – cz. I



- Wyznaczenie okręgu centralnego dłoni (około 95% jego powierzchni powinny zajmować piksele dłoni).

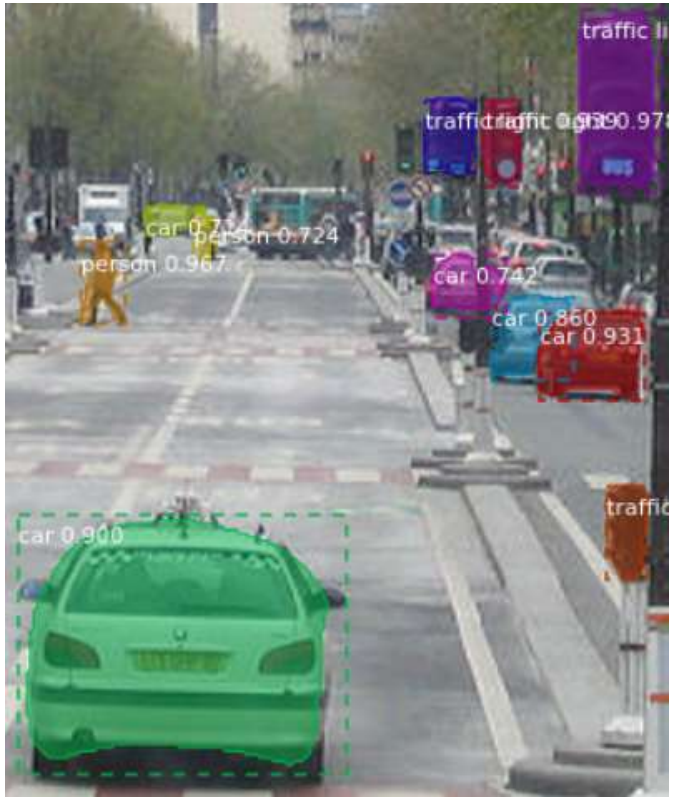
- Operacja flood fill w punkcie startowym F.



- [opcjonalnie]
  Obrócenie dłoni do pionu, czyli wykonanie takiego obrotu, żeby wektor o początku w punkcie C i końcu w punkcie najodleglejszym od C był skierowany ku górze.

Segmentacja przy użyciu sieci głębokich

- Istnieją sieci głębokie, które są w stanie dokonać detekcji obiektów na obrazie. Są to np. sieci: YOLO, DETR, R-CNN, Mask R-CNN.
- Ostatnia z wymienionych sieci nadaje się do problemu segmentacji obiektów, ponieważ wyznacza nie tylko prostokątny obszar, w którym mieści się szukany obiekt, ale również zaznacza piksele wewnątrz prostokąta, które należą do obiektu.
- Można więc próbować wykorzystać sieć Mask R-CNN do segmentacji dłoni po nauczeniu/douczeniu na odpowiednich danych.



Źródło: *https://github.com/matterport/Mask_RCNN*

Konwersja mapy głębi na chmurę punktów

Współrzędne punktów chmury PCh_i^x, PCh_i^y i PCh_i^z ustawiane są w zależności od współrzędnych pikseli mapy głębi DA_i^x i DA_i^y oraz parametrów wewnętrznych kamery

$$PCh_i^x = \frac{(DA_i^z + fl) \cdot (\frac{DA_{width}}{2} - DA_i^x - 1) \cdot ps_x}{fl}$$

$$PCh_i^y = \frac{(DA_i^z + fl) \cdot (\frac{DA_{height}}{2} - DA_i^y - 1) \cdot ps_y}{fl}$$

$$PCh_i^z = DA_i^z$$

gdzie DA_{width} i DA_{height} są liczbami odpowiednio kolumn i wierszy mapy głębi DA; fl jest długością ogniskowej kamery; ps_x i ps_y oznaczają wymiary piksela, odpowiednio długość i szerokość.

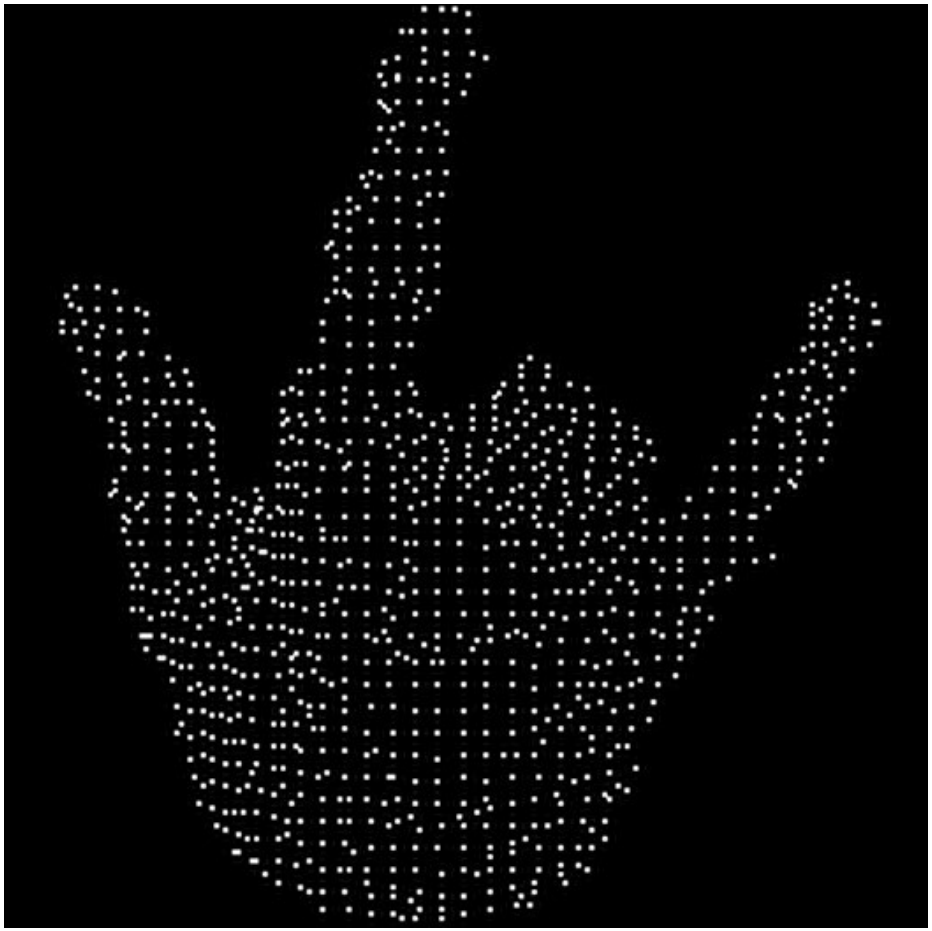
Konwersja mapy głębi na chmurę punktów – parametry wewnętrzne kamery

Wartości parametrów wewnętrznych kamer Kinect i Kinect 2 są następujące:

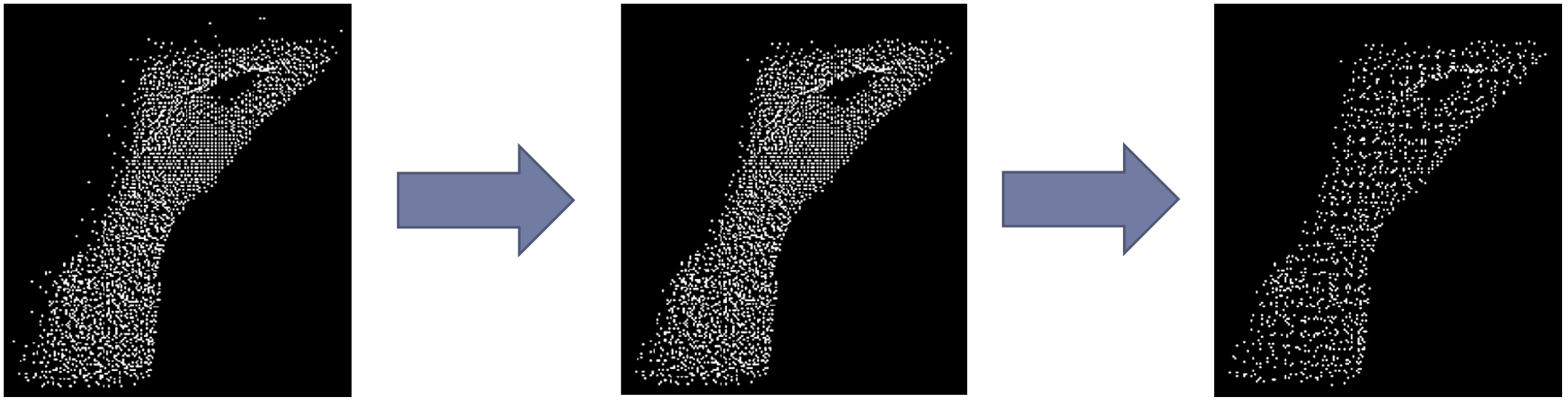
- Kinect: $fl = 4.73 \ \text{mm}, ps_x = ps_y = 0.0078 \ \text{mm}$
- Kinect 2: $fl = 3.657 \ \text{mm}, ps_x = ps_y = 0.01 \ \text{mm}$

Wartości te zostały wyliczone poprzez kalibrację kamer z wykorzystaniem oprogramowania Camera Calibration Toolbox for Matlab.

Mapa głębi dłoni w formie chmury punktów (wynik dotychczasowych operacji)



Filtracja i downsampling chmury punktów (krok opcjonalny)



Filtr usuwający punkty izolowane, czyli takie, które w zadanym promieniu mają mniej niż określoną liczbę sąsiadów. Implementacja w bibliotece PCL - klasa *RadiusOutlierRemoval*, metoda *applyFilter*.

Downsampling – zmniejszenie liczby punktów chmury. Implementacja w bibliotece PCL - klasa *VoxelGrid*, metoda *applyFilter*.

Wyznaczanie cech chmur punktów – przykładowe deskryptory

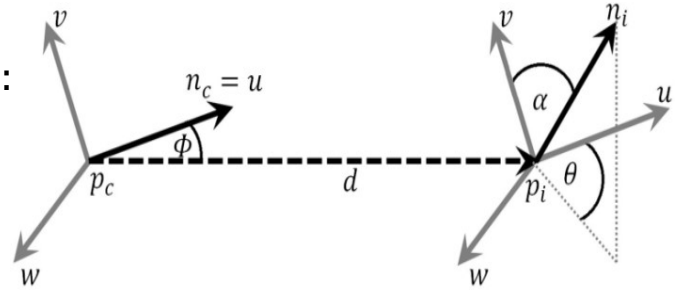
- VFH – Viewpoint Feature Histogram
- GRSD – Global Radius-based Surface Descriptor
- ESF – Ensemble of Shape Functions

Strona internetowa z opisem, przykładami i źródłami powyższych (oraz innych) deskryptorów chmur punktów:

https://robotica.unileon.es/index.php?title=PCL/OpenNI_tutorial_4:_3D_object_recognition_(descriptors)

VFH – Viewpoint Feature Histogram

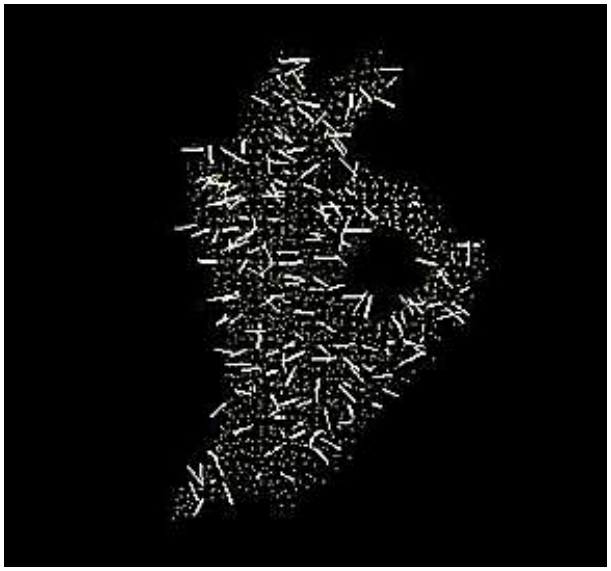
- Opiera się na analizie wektorów normalnych do powierzchni rozpiętej na chmurze.
- Składa się z dwóch komponentów:
  - kształtu powierzchni (4 histogramy) - rozkład następujących wartości:
    - θ - kąt yaw między wektorami,
    - $\cos(\alpha)$ - cosinus kąta pitch między wektorami,
    - $\cos(\phi)$ - cosinus kąta między pierwszym wektorem (n_c) a linią łączącą go z punktem przyłożenia drugiego wektora (n_i),
    - d - odległość między punktami przyłożenia wektorów.
  - kierunku patrzenia (1 histogram) - rozkład kątów φ między każdym wektorem a kierunkiem patrzenia.



Źródło: *https://doi.org/10.1109/IROS.2010.5651280*

VFH – Viewpoint Feature Histogram

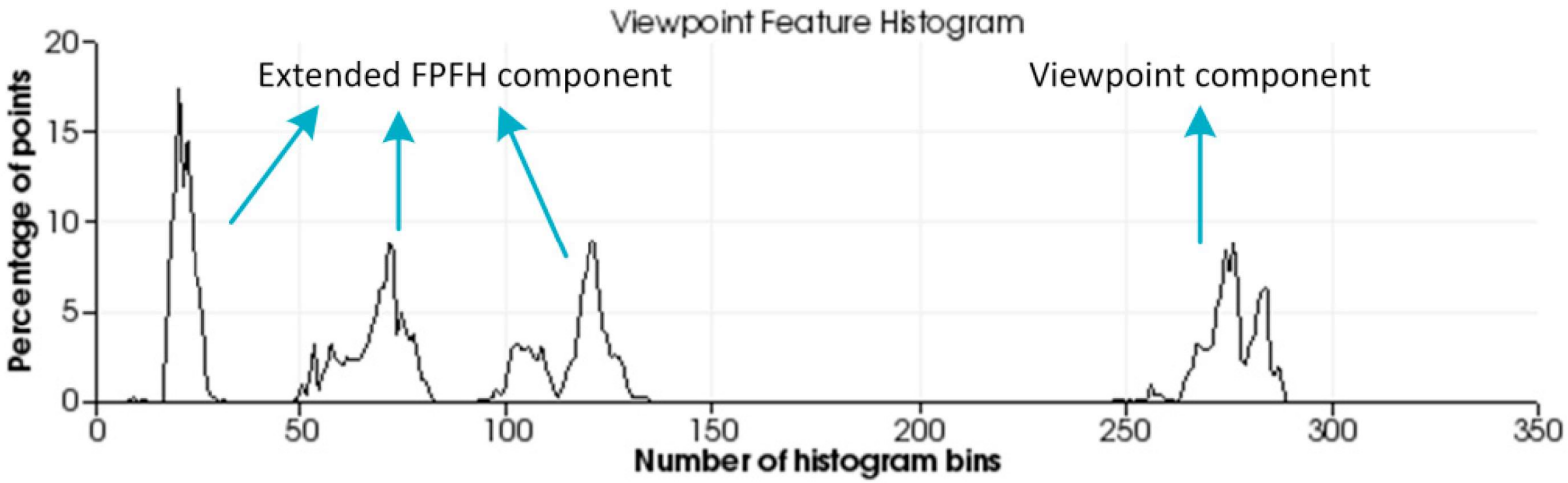
- Komponent kształtu powierzchni
  - Może być użyty do kategoryzowania (rozpoznawania) kształtu obiektów 3D, np. układów dłoni (niezależnie od ich orientacji).
- Komponent kierunku patrzenia
  - Można go użyć w celu ustalenia orientacji obiektów 3D względem kamery.



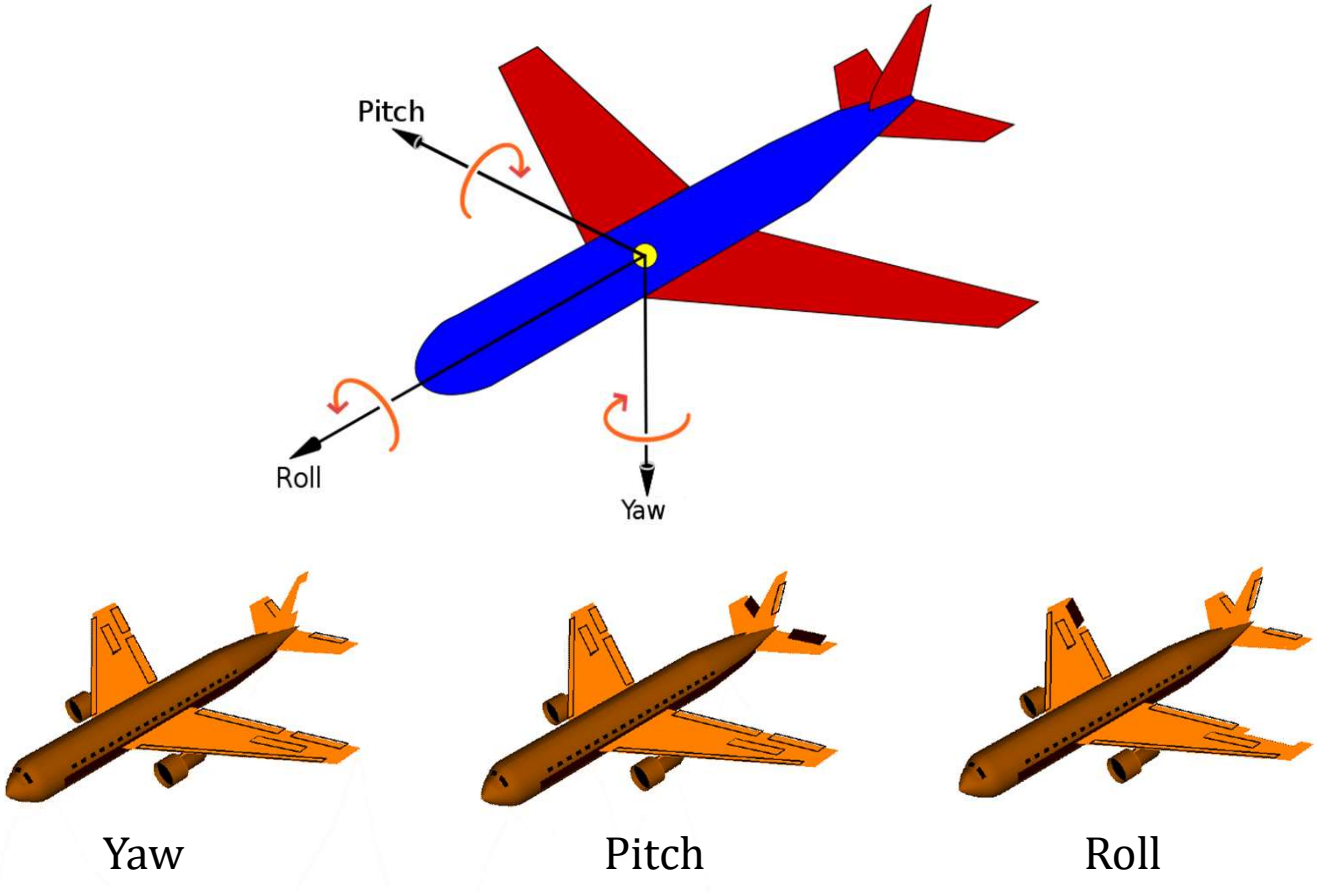
Chmura punktów z wektorami normalnymi (prostopadłymi) do powierzchni

VFH – Viewpoint Feature Histogram - histogram

Na koniec, obliczone cechy obu komponentów są grupowane w formie histogramu.



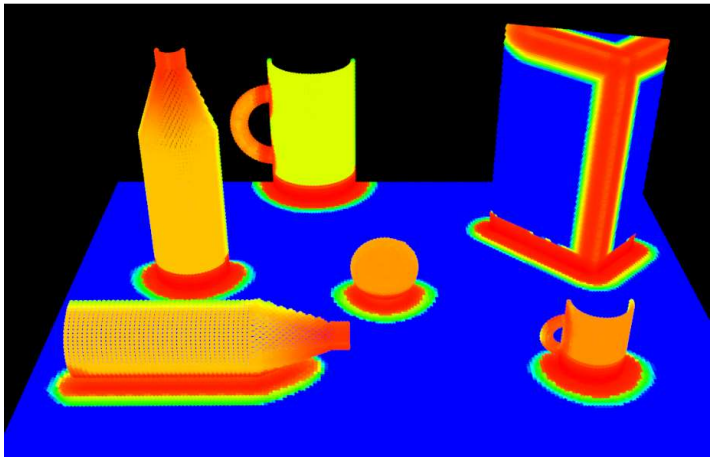
Kąty yaw (myszkowanie) pitch (odchylenie) i roll (turlanie)



Źródło: https://en.wikipedia.org/wiki/Aircraft_principal_axes

GRSD – Global Radius-based Surface Descriptor

- Opisuje radialne relacje punktów z ich sąsiedztwem.
- Jeden histogram - 21 binów.



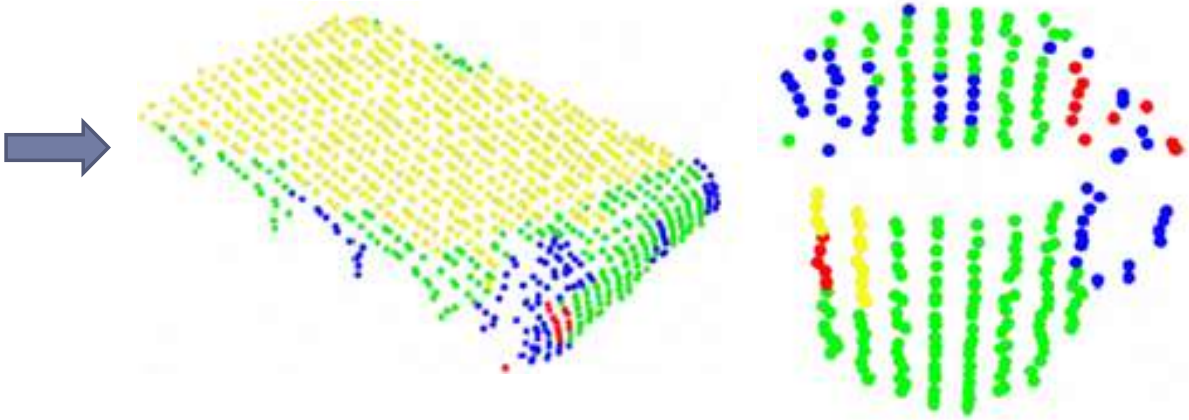
Obraz składający się z wartości RSD. Kolory cieplejsze oznaczają bardziej zakrzywione powierzchnie.

Wartości RSD – promienie sfer wpasowanych w powierzchnię wokół danego punktu.

Źródło: *https://doi.org/10.1109/ICHR.2010.5686323*

GRSD – kategoryzacja powierzchni

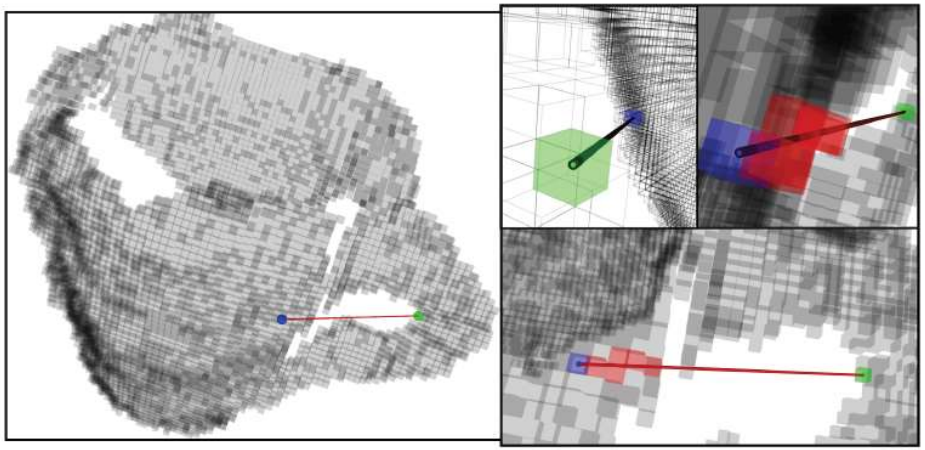
Algorytm CRF (Conditional Random Field)
+ wartości RSD jako dane wejściowe



Kategorie powierzchni:

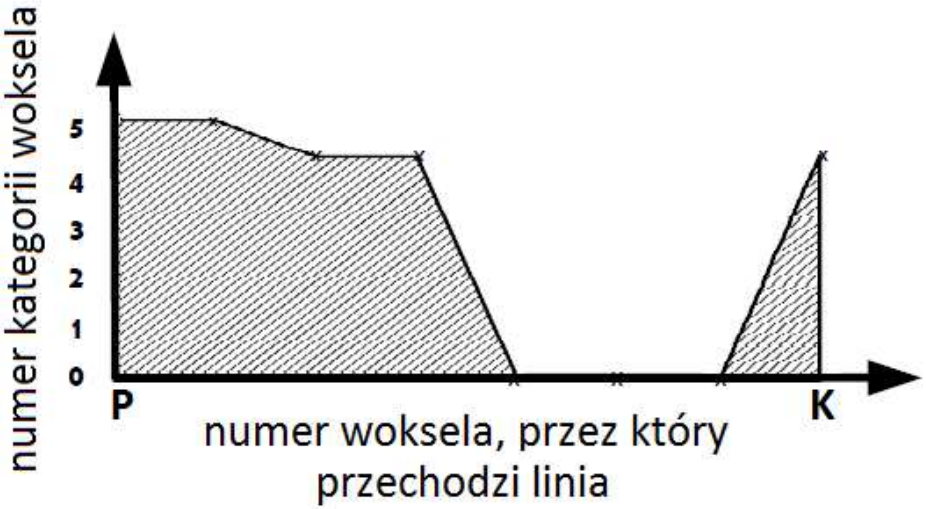
1. płaszczyzna (żółty),
2. walec (zielony),
3. ostra krawędź lub szum (czerwony),
4. obrzeże – np. granice, przejścia pomiędzy powierzchniami (niebieski),
5. sfera (nie występuje na rysunkach).

GRSD – tworzenie histogramu



Wyznaczanie GRSD na siatce wokseli (voxel grid). Rysunek przedstawia przykładową linię łączącą woksele.

Wykres przecinania się linii z wokselami
P – woksel początkowy
K – woksel końcowy



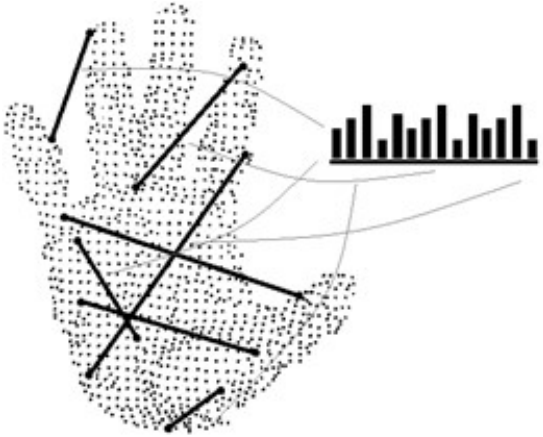
ESF – Ensemble of Shape Functions

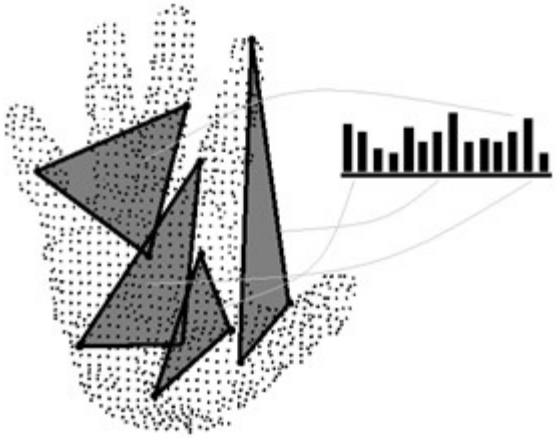
- Stanowi kombinację trzech funkcji kształtu chmury punktów, opisujących: odległości, kąty i obszary.
- 10 histogramów po 64 biny.
- Metoda liczenia histogramów:
  - nie wymaga wyznaczania wektorów normalnych.
  - jest niedeterministyczna – wielokrotnie losowane są trójki punków.

Źródło: *https://doi.org/10.1109/ROBIO.2011.6181760*

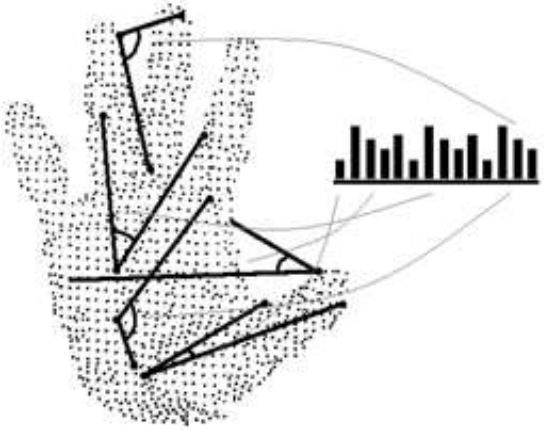
ESF – cechy

D2 – odległości pomiędzy punktami



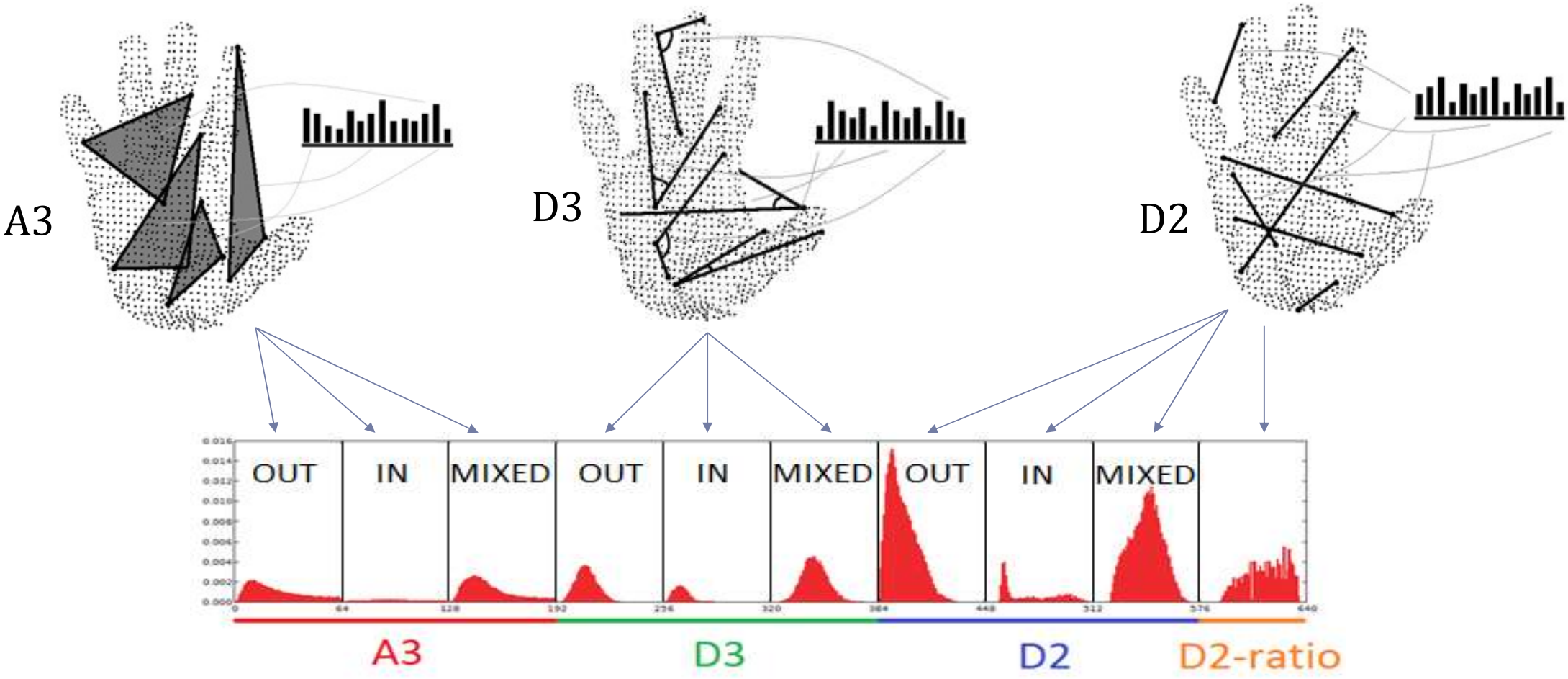


D3 – powierzchnia trójkątów wyznaczonych przez punkty

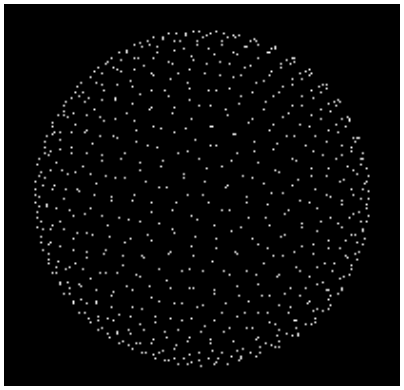


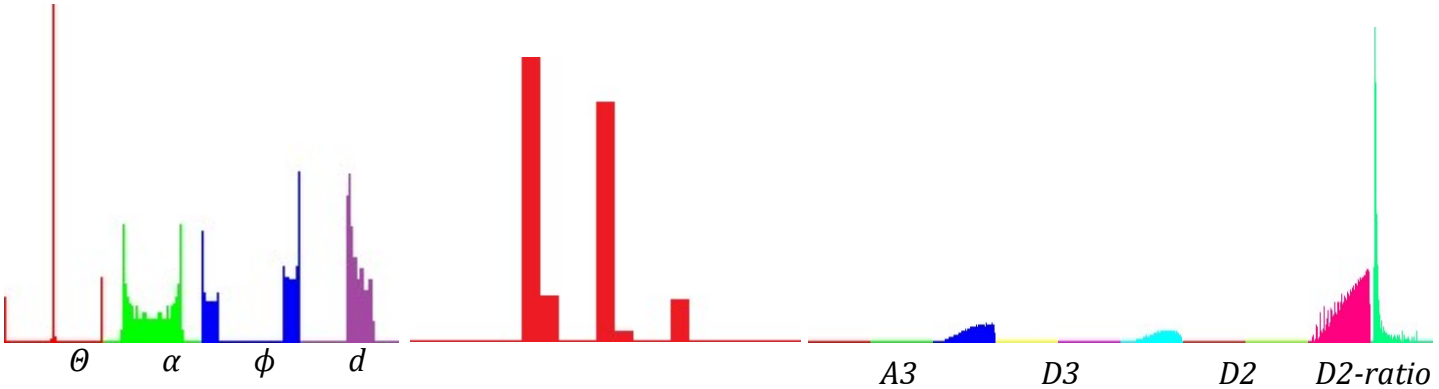
A3 – kąty pomiędzy liniami utworzonymi z połączeń punktów

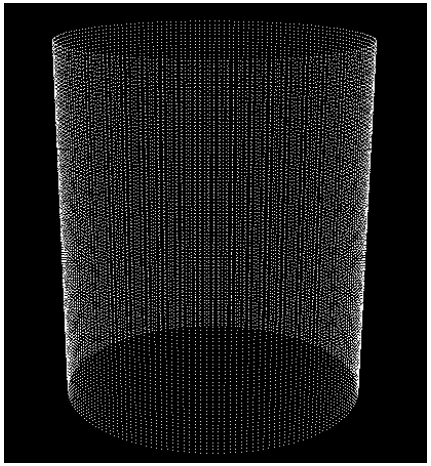
ESF - cechy

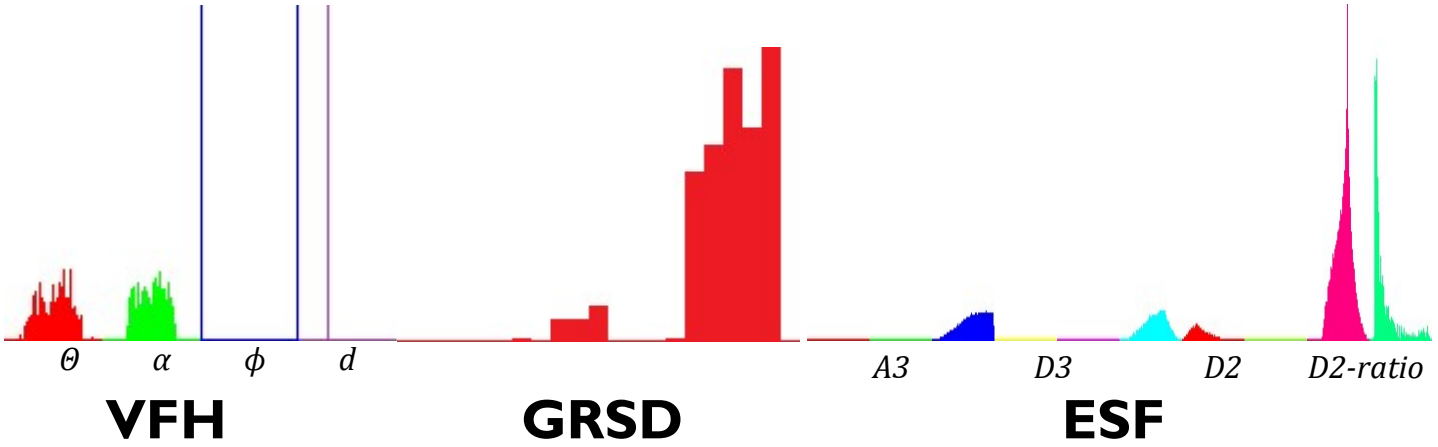


VFH, GRSD i ESF dla brył elementarnych

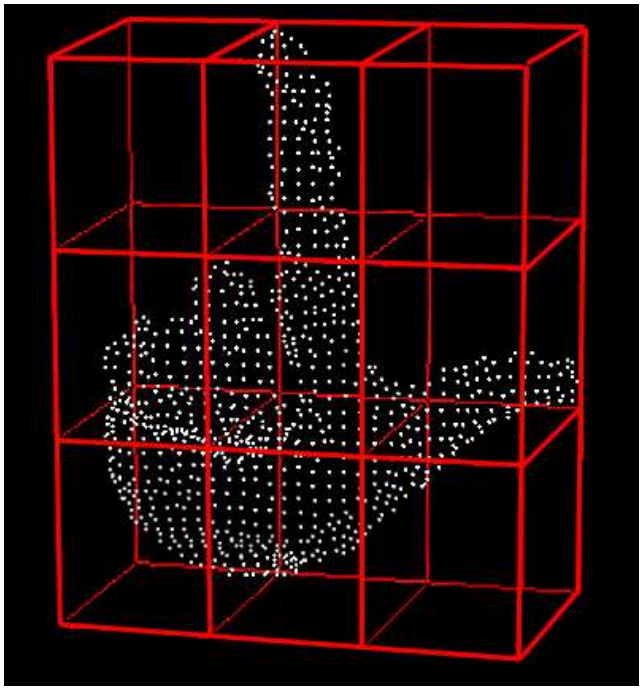




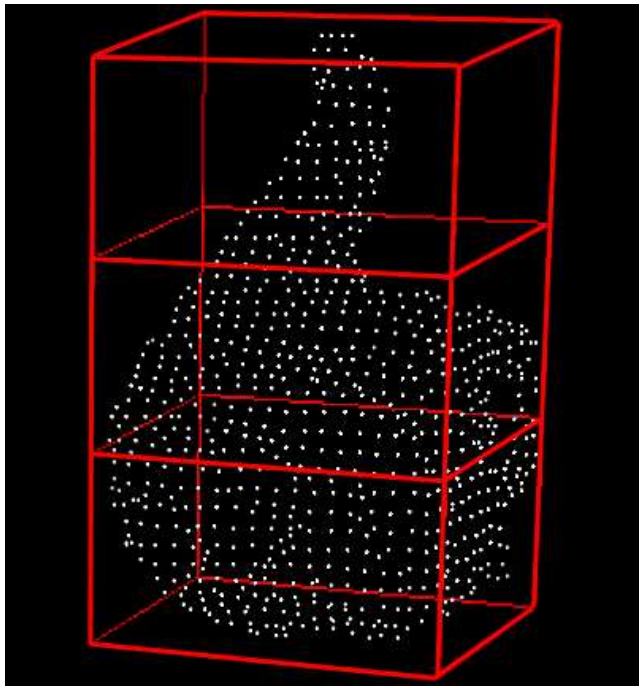




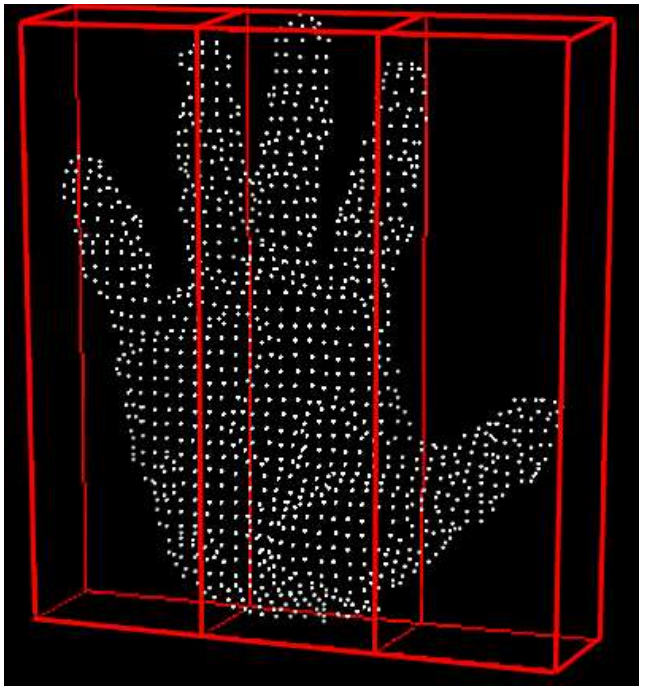
Podział chmury punktów na komórki



9 komórek



3 horyzontalne komórki



3 wertykalne komórki

Wektory cech

- Po obliczeniu deskryptorów musimy utworzyć wektory cech, czyli zestawy wartości, na podstawie których klasyfikator będzie rozpoznawał gest. Wektory mogą się składać z:
  1. wszystkich wartości wybranych histogramów;
  2. obliczonych statystyk dla każdego histogramu: średnia, odchylenie standardowe (ewentualnie mediana).

- Pierwsze podejście wydaje się bardziej precyzyjne, ale drugie znacznie zmniejszy liczbę cech, co bardzo wpłynie na szybkość uczenia i klasyfikacji. Może też zmniejszyć prawdopodobieństwo przeuczenia klasyfikatora, ponieważ cechy są uogólnione.

Wektory cech

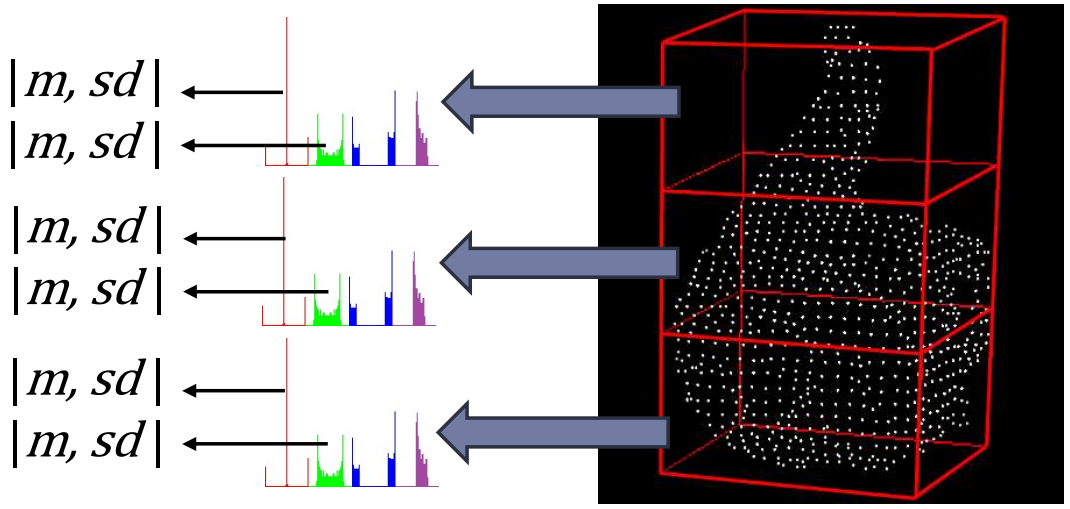
Przykładowy wektor cech w przypadku podziału bryły brzegowej na 3 komórki.

Wybrane histogramy VFH: ϕ i d reprezentowane są przez średnią i odchylenie standardowe histogramów.

Komórka I				Komórka II				Komórka III			
$m(\phi)$	$sd(\phi)$	$m(d)$	$sd(d)$	$m(\phi)$	$sd(\phi)$	$m(d)$	$sd(d)$	$m(\phi)$	$sd(\phi)$	$m(d)$	$sd(d)$

m – średnia

sd – odchylenie standardowe



Normalizacja cech (min-max)

Wszystkie cechy klasyfikowanych próbek oraz występujących w zbiorze treningowym muszą być znormalizowane przed podaniem ich do klasyfikatora.

Najlepiej normalizację przeprowadzić za pomocą skalowania każdej cechy *x* do zakresu [0-1]:

$$x' = \frac{(x - X_{min}) \cdot (Y_{max} - Y_{min})}{(X_{max} - X_{min})}$$

gdzie:
x – wartość oryginalna cechy,
x' – wartość znormalizowana cechy,
Xmin, *Xmax* – minimalna i maksymalna wartość cechy występująca w całym zbiorze treningowym,
Ymin, *Ymax* – minimalna i maksymalna wartość przedziału docelowego (w naszym przypadku odpowiednio 0 i 1).

Normalizacja typu z-score (zero-mean)

Alternatywna metoda normalizacji, której wynikiem są wartości cech „rozrzucone” wokół zera (ujemne oraz dodatnie). Ich średnia jest równa 0, a odchylenie standardowe 1.

$$x' = \frac{x - mean}{stdev}$$

gdzie:
x – wartość oryginalna cechy,
x' – wartość znormalizowana cechy,
mean – średnia wartość cechy w zbiorze treningowym,
stdev – odchylenie standardowe wartości cechy w zbiorze treningowym.

Która metoda normalizacji jest lepsza: min-max, czy z-score?

- Niestety nie ma na to pytanie jednoznacznej odpowiedzi. To, która metoda będzie skuteczniejsza może zależeć od danych (rozkładów wartości cech) oraz od klasyfikatora. Teoretycznie:
  - jeśli dane mają rozkład daleki od normalnego, lepsza będzie normalizacja min-max;
  - jeśli dane mają dużo punktów odstających, lepsza będzie normalizacja z-score.

- Jeśli zależy nam na jak największej dokładności klasyfikacji, to warto przetestować obie metody normalizacji.

- Należy jednak pamiętać, że próbki klasyfikowane powinny być znormalizowane tą samą metodą, która została zastosowana do zbioru treningowego.

Klasyfikacja

Proponowane klasyfikatory:

- **k-najbliższych sąsiadów** (ang. k-Nearest Neighbors, **kNN**) – opis od slajdu 33.
- **maszyna wektorów nośnych** (maszyna wektorów wspierających, Support Vector Machine, **SVM**) – opis od slajdu 36.
- **probabilistyczna sieć neuronowa** (ang. Probabilistic Neural Network, **PNN**) oraz inne rodzaje sieci neuronowych (płytkich i głębokich).
- **drzewa decyzyjne** (ang. decision trees) – klasyfikacja opiera się na sprawdzaniu wyuczonych reguł zapisanych na strukturze drzewa.
- **losowe lasy decyzyjne** (ang. random forests) – tworzenie (z elementami losowości) wielu różnych drzew decyzyjnych; klasyfikacja polega na wyznaczeniu etykiety najczęściej powtarzającej się klasy wśród etykiet zwróconych przez każde drzewo.

Przykładowe implementacje proponowanych klasyfikatorów w Matlabie

- **k-najbliższych sąsiadów (kNN)** – funkcja *knnsearch* i toolbox Classification Learner.
- **maszyna wektorów nośnych (SVM)** – toolbox Classification Learner lub biblioteka libSVM (*https://www.csie.ntu.edu.tw/~cjlin/libsvm*).
- **probabilistyczna sieć neuronowa (PNN)–** funkcja *newpnn.*
- **drzewa decyzyjne** – funkcja *fitctree* i toolbox Classification Learner.
- **losowe lasy decyzyjne** – toolbox Classification Learner (różne warianty, np. Bagged Trees).

Przykładowe implementacje proponowanych klasyfikatorów w Pythonie

- Większość spośród wymienionych klasyfikatorów dostępna jest w module *sklearn*.

- Wyjątkiem jest sieć PNN. Jest ona jednak dostępna w innym module:
  - *https://github.com/vdevmcitylp/probabilistic-neural-network*

Klasyfikator k-najbliższych sąsiadów (ang. k-nearest neighbors, kNN)

- Najbardziej powszechny i najprostszy w implementacji klasyfikator.
- Algorytm kNN jest następujący:
  1. Oblicz odległość (w ustalonej metryce) próbki klasyfikowanej od wszystkich próbek ze zbioru treningowego.
  2. Znajdź *k* próbek, których obliczona odległość jest najmniejsza (tzw. najbliższych sąsiadów).
  3. Jako wynik klasyfikacji zwróć etykietę klasy, która najczęściej powtarza się wśród *k* sąsiadów.
  4. Jeśli nie można ustalić jednoznacznie zwycięskiej klasy (np. dwóch najbliższych sąsiadów należy do klasy A i dwóch do klasy B), to wybierz klasę próbki z najmniejszą odległością (spośród próbek A i B).

Klasyfikator k-najbliższych sąsiadów – najczęściej używane metryki

- Metryka euklidesowa (ang. Euclidean):

$$D_{euk}(x, y) = \sqrt{\sum_{i=0}^{N}(x_i - y_i)^2}$$

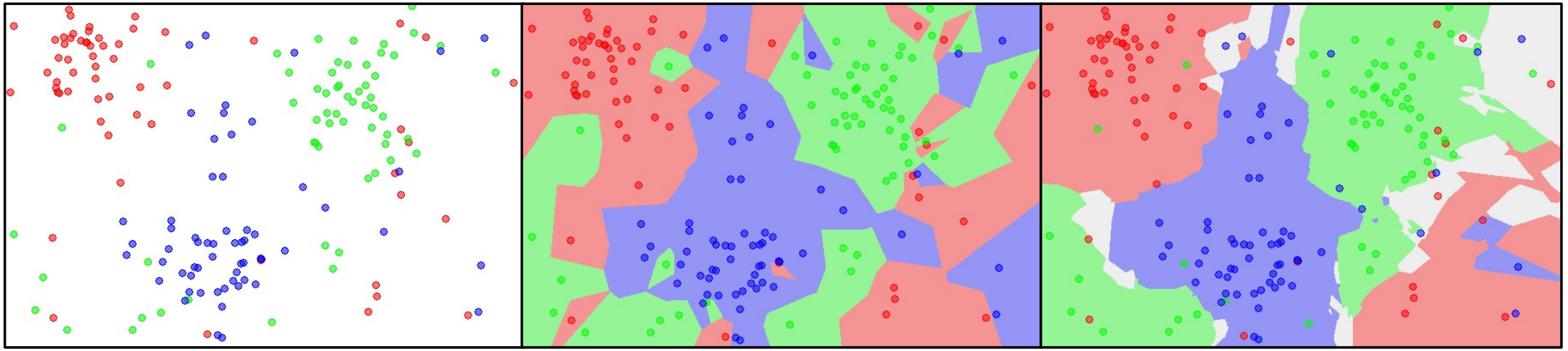
W celu przyspieszenia obliczeń można pominąć pierwiastek. Jeśli używamy standardowego kNN (bez metod takich jak DTW), to wyniki klasyfikacji uzyskane bez pierwiastka powinny być identyczne).

- Metryka miejska (taksówkowa, ang. city-block, Manhattan):

$$D_{cb}(x, y) = \sum_{i=0}^{N} |x_i - y_i|$$

gdzie N jest liczbą cech odpowiednio obiektów x i y.

Klasyfikator k-najbliższych sąsiadów – mapy klasyfikacji



próbki należące do trzech klas | mapa klasyfikacji 1NN | mapa klasyfikacji 5NN

Źródło: *https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm*

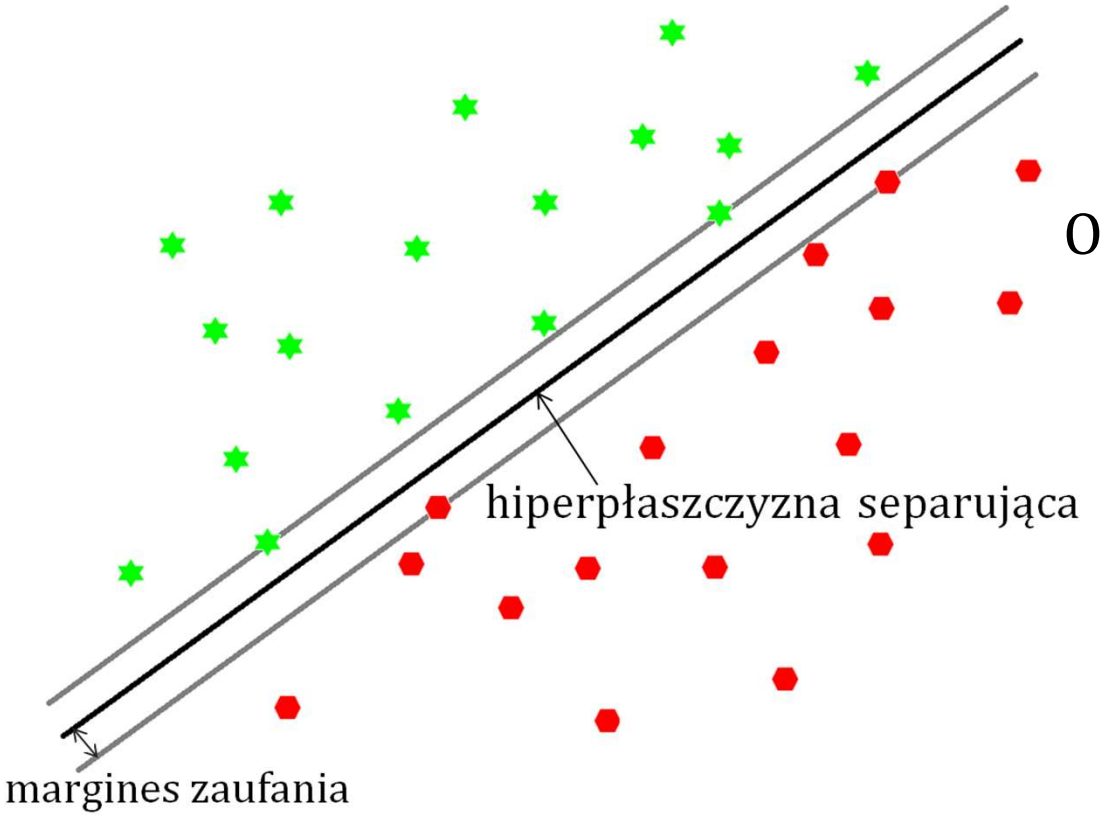
Maszyna wektorów nośnych (suport vector machine, SVM)

- SVM, to Klasyfikator binarny (nadający się do problemów dwuklasowych).
- Istnieje jednak możliwość zastosowania go do klasyfikacji wieloklasowej (slajdy 36-40).

Ogólna zasada działania:

1. Obserwacja próby poddawana jest nieliniowej transformacji z oryginalnej przestrzeni cech do przestrzeni wyższego wymiaru.
2. W procesie uczenia wyznaczana jest hiperpłaszczyzna separująca dane obu klas. Hiperpłaszczyzna jest opisana funkcją liniową. Jest to możliwe dzięki transformacji przeprowadzonej w pierwszym punkcie.

Maszyna wektorów nośnych – hiperpłaszczyzna separująca



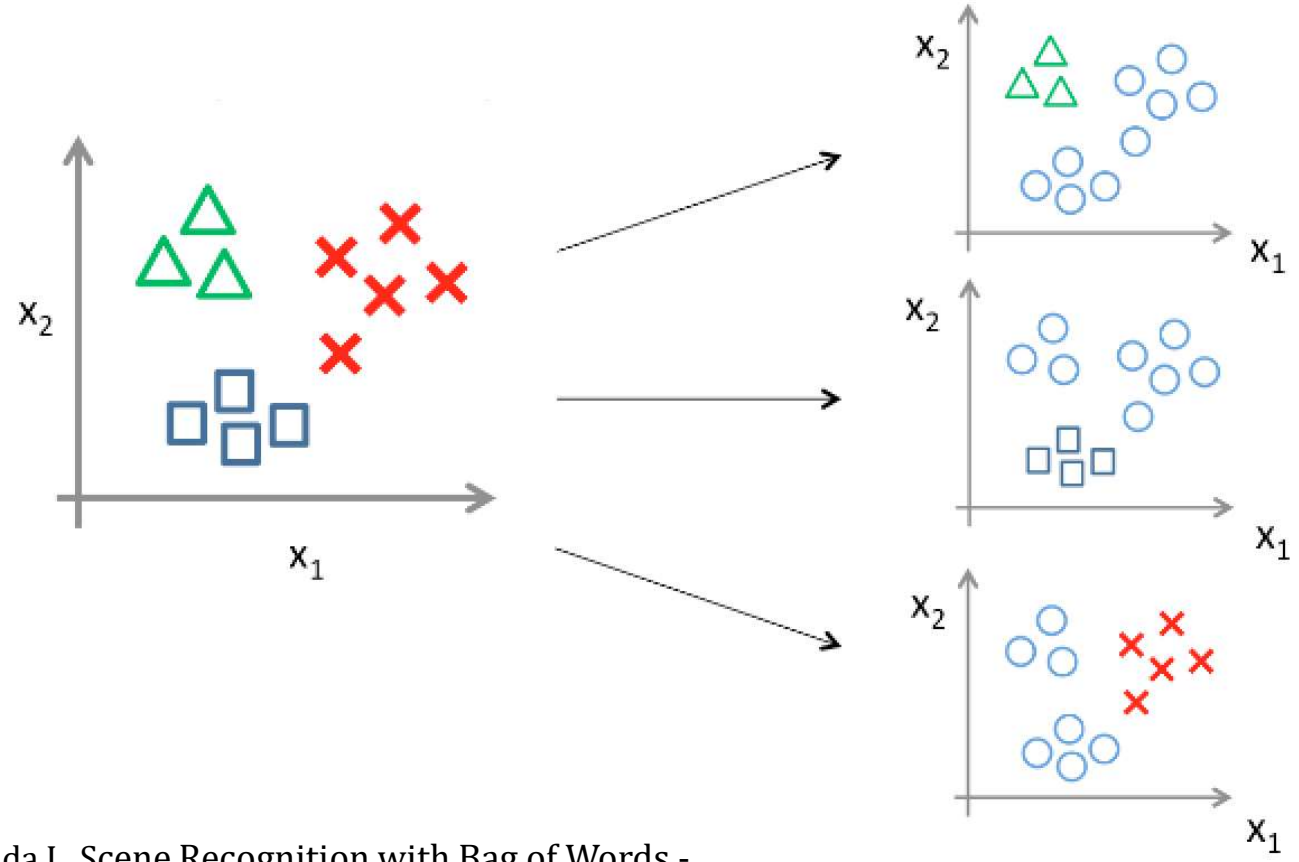
Ogólna zasada działania (c.d.):

3. Hiperpłaszczyzna musi uwzględniać możliwie jak największy margines zaufania, czyli odległość hiperpłaszczyzny od najbliższej obserwacji ze zbioru treningowego.
4. W procesie klasyfikacji próbka zostaje zakwalifikowana do jednej z dwóch klas w zależności od tego, po której stronie hiperpłaszczyzny się znajduje.

Maszyna wektorów nośnych – klasyfikacja wieloklasowa one-versus-all

- Tworzy się *n* klasyfikatorów binarnych, gdzie *n* – liczba klas.
- Uczenie: etykietę „1” przypisuje się klasie *k*, natomiast etykietę „0” – wszystkim pozostałym klasom.
- Klasyfikacja: klasyfikatory binarne zwracają wartości prawdopodobieństwa *p* przynależności klasyfikowanej próbki do klasy *k*.
- Wynikiem klasyfikacji jest klasa, dla której prawdopodobieństwo *p* jest największe.

One-versus-all - wizualizacja

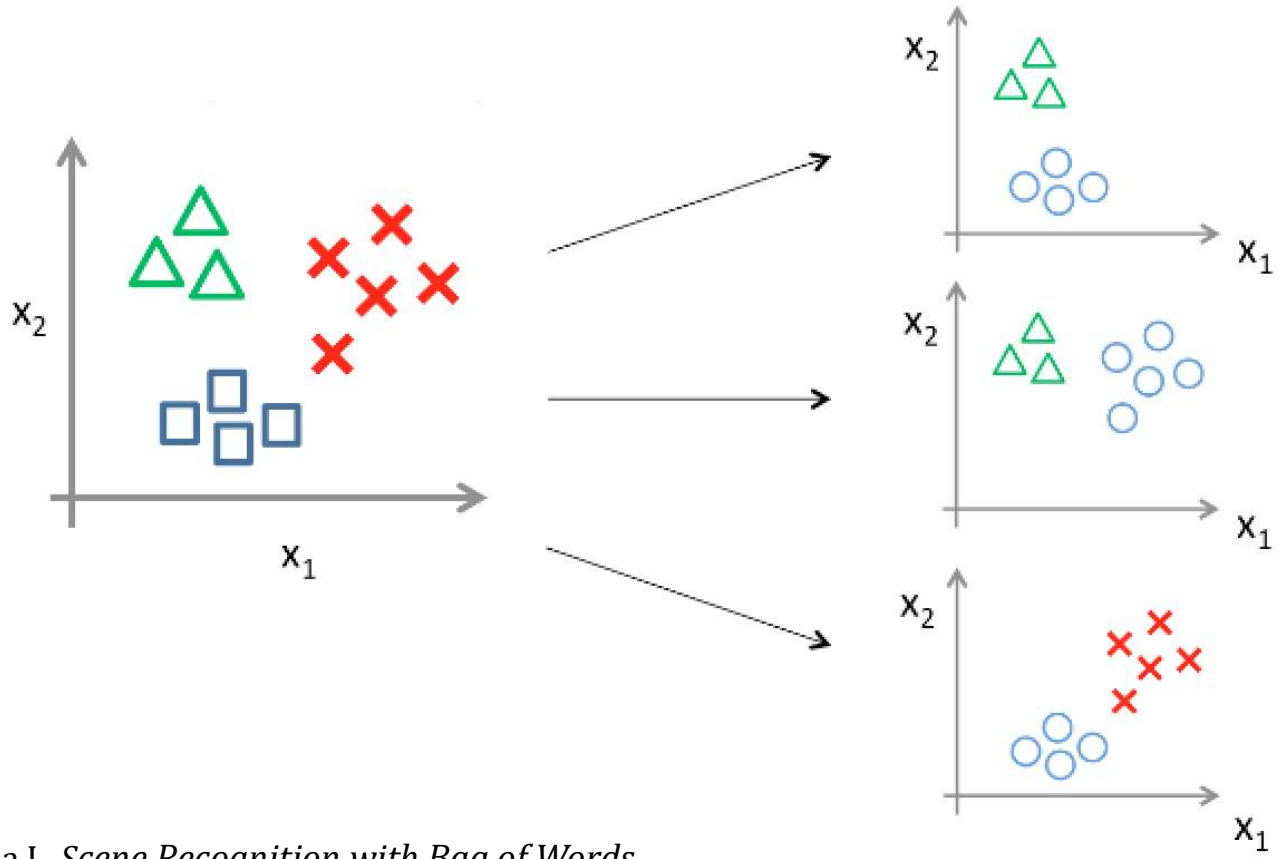


Na podstawie: Nanda J., Scene Recognition with Bag of Words - *https://www.cc.gatech.edu/classes/AY2016/cs4476_fall/results/proj4/html/jnanda3/index.html*

Maszyna wektorów nośnych – klasyfikacja wieloklasowa one-versus-one (zwana również all-versus-all)

- Tworzy się $n(n\text{-}1)/2$ klasyfikatorów binarnych, gdzie n – liczba klas.
- Uczenie: etykietę „1” przypisuje się klasie k, natomiast etykietę „0” – jednej z pozostałych klas. Każdy klasyfikator binarny dotyczy innej pary klas.
- Klasyfikacja: klasyfikatory binarne zwracają wartości prawdopodobieństwa p przynależności klasyfikowanej próbki do klasy k.
- Wynikiem klasyfikacji jest klasa, dla której łączna suma prawdopodobieństw p jest największa.

One-versus-one (all-versus-all) - wizualizacja



Na podstawie: Nanda J., *Scene Recognition with Bag of Words - https://www.cc.gatech.edu/classes/AY2016/cs4476_fall/results/proj4/html/jnanda3/index.html*

Która metoda klasyfikacji wieloklasowej jest lepsza?

- One-versus-all - jest szybsza i łatwiejsza w implementacji
- One-versus-one (all-versus-all) – najczęściej wiąże się z dokładniejszą klasyfikacją

Walidacja metod rozpoznawania (klasyfikacji)

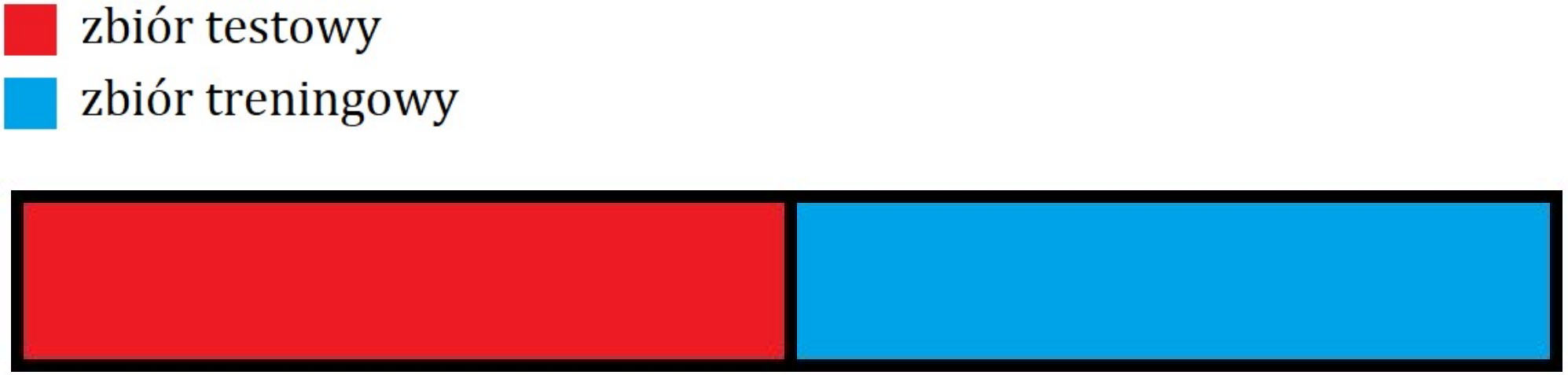
- Walidacja jest sposobem weryfikacji efektywności algorytmu rozpoznawania.
- Polega na podziale zbioru danych na zbiór testowy i treningowy (trenujący, uczący).
- Wszystkie próbki ze zbioru testowego są klasyfikowane metodą wyuczoną na podstawie próbek zbioru treningowego. Następnie zlicza się procent prawidłowych klasyfikacji (dokładność).
- Próbki występujące w zbiorze treningowym **nie mogą** występować w zbiorze testowym.

Najczęściej stosowane rodzaje walidacji:

- walidacja 50-50
- k-krotna walidacja krzyżowa (k-fold cross-validation)
- walidacja typu LOSO (leave one subject out)

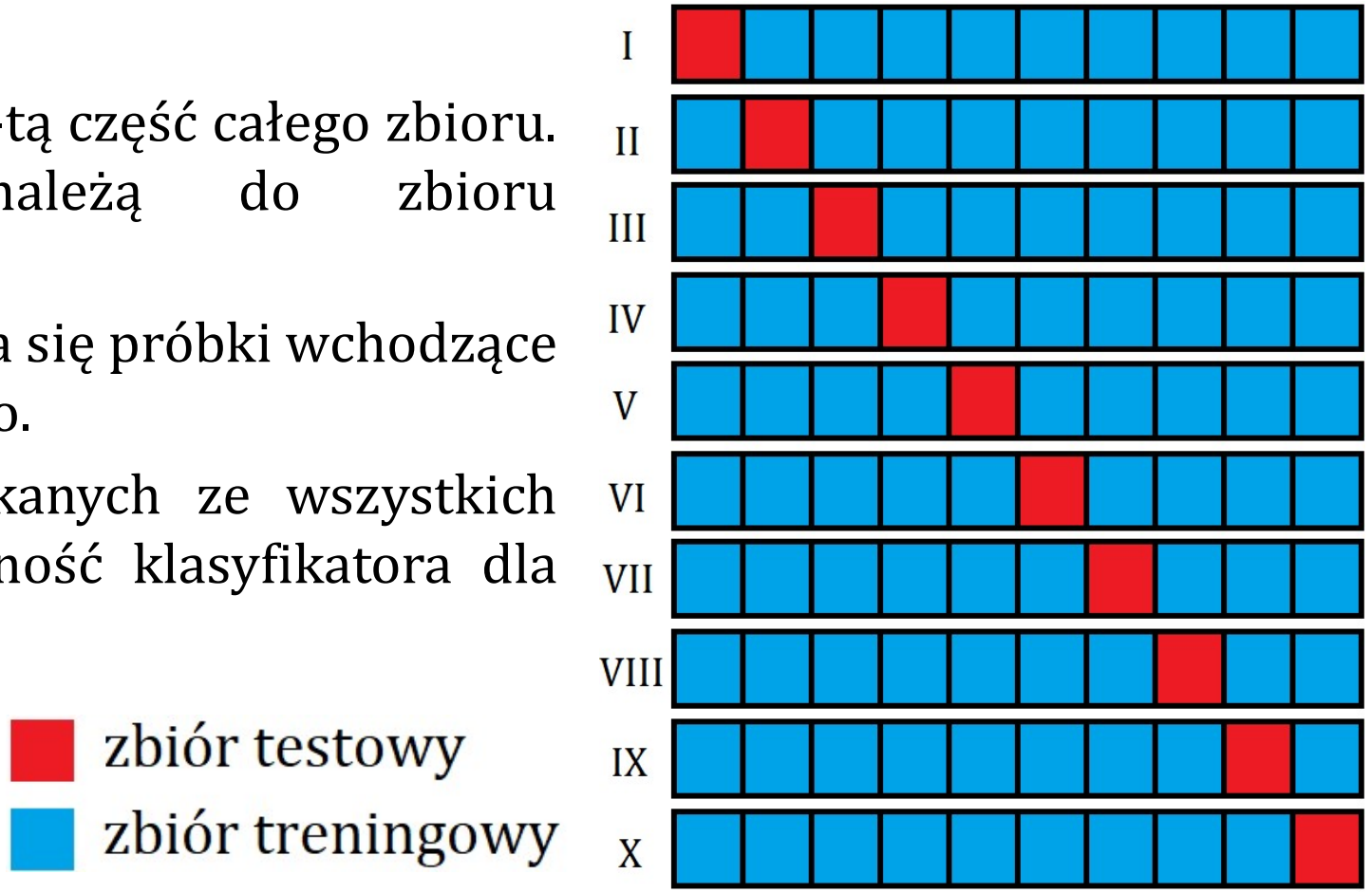
Walidacja 50-50

- Wykonuje się jeden test.
- Zbiór testowy i treningowy są dzielone na dwie równe części.
- Wariacją tej metody jest walidacja typu holdout (ang. holdout validation), w której określa się rozmiar zbioru testowego inny niż 50% całości.



k-krotna walidacja krzyżowa (ang. k-fold cross-validation)

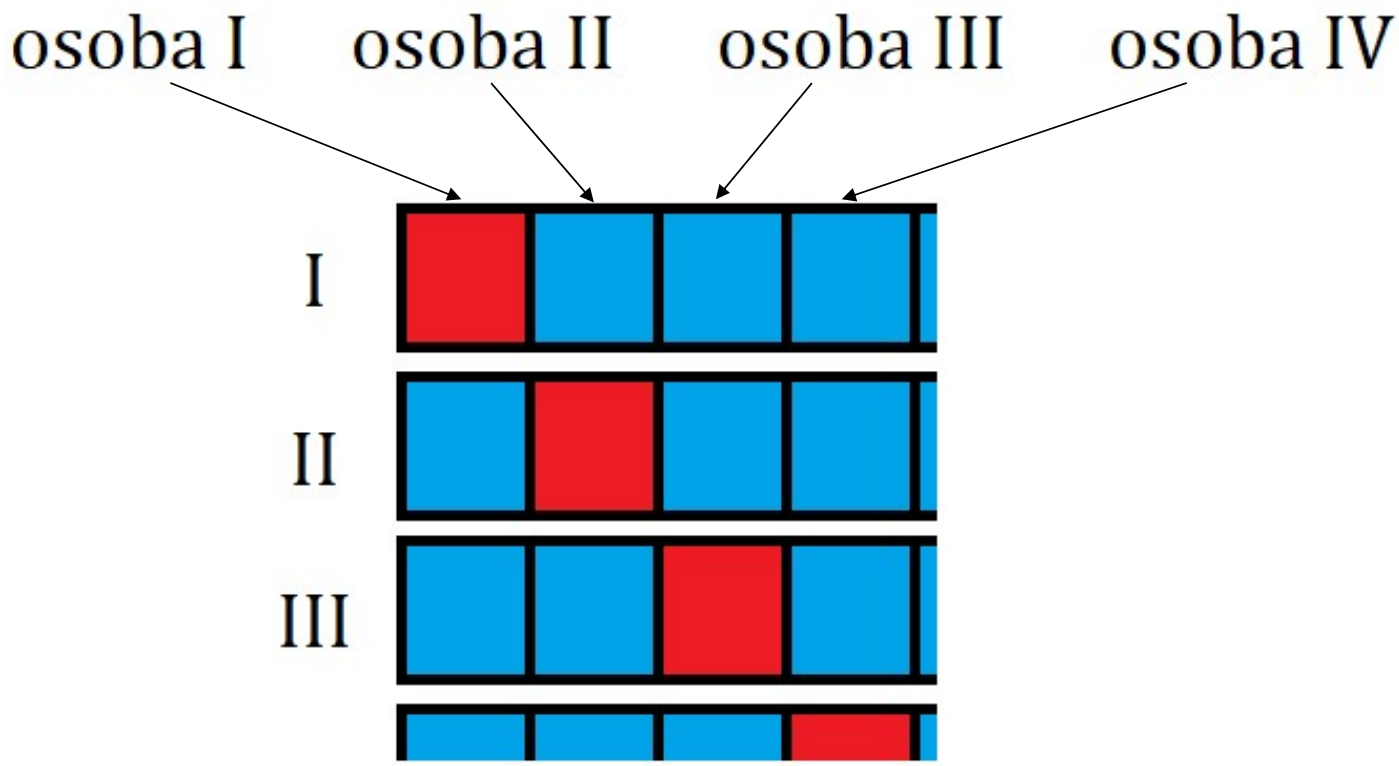
- Wykonuje się *k* testów.
- Zbiór testowy stanowi *k*-tą część całego zbioru. Pozostałe próbki należą do zbioru treningowego.
- W każdym teście zmienia się próbki wchodzące w skład zbioru testowego.
- Średnia wyników uzyskanych ze wszystkich testów opisuje efektywność klasyfikatora dla całego zbioru danych.



Walidacja typu LOSO (leave-one-subject-out)

- Szczególny przypadek k-krotnej walidacji krzyżowej.
- W każdym z *k* testów próbki zbioru testowego reprezentują gesty wykonywane przez osoby nieobecne w zbiorze treningowym.
- Liczba *k* powinna być równa liczbie osób wykonujących gesty.
- W wariacji metody LOSO możemy zastosować liczbę *k* mniejszą, ale podzielną przez całkowitą liczbę osób (np. leave-two-subjects-out).
- Najtrudniejszy rodzaj walidacji i jednocześnie najbardziej reprezentatywny.
- Symuluje działanie algorytmu w warunkach rzeczywistych (osoba, której gesty są rozpoznawane najczęściej nie uczestniczyła w przygotowywaniu zbioru treningowego).

Walidacja typu LOSO (leave-one-subject-out)



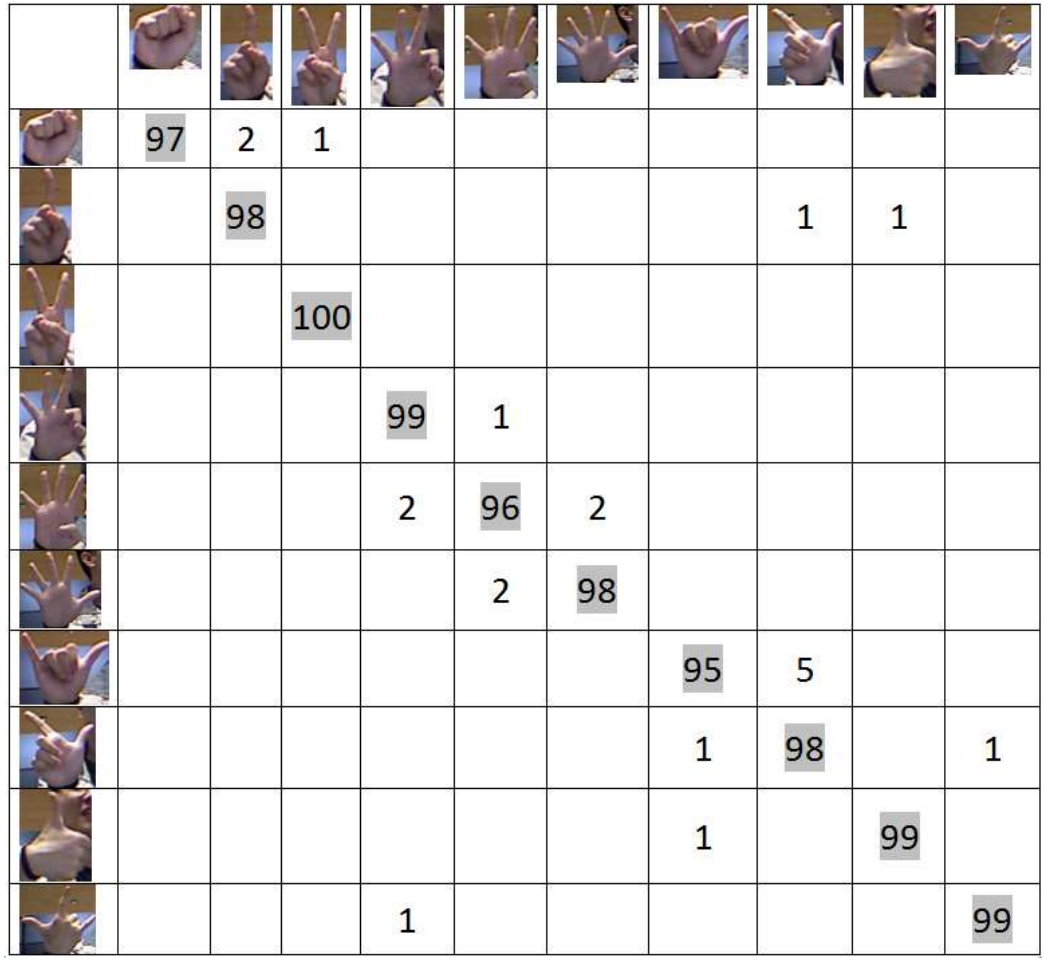
Testy walidacji w przypadku metod z elementami losowości

- Jeśli metoda rozpoznawania obiektów ma na jakimkolwiek etapie elementy losowości, to całą walidację należy powtórzyć wielokrotnie i obliczyć średnią oraz odchylenie standardowe z wyników.
  - Np. 10-krotną walidację krzyżową powtarzamy 10 razy, czyli wykonujemy łącznie 100 testów.

- Liczba powtórzeń powinna być większa w przypadku, gdy wyniki są bardziej rozrzucone, tzn. gdy jest większe odchylenie standardowe.

Tablica pomyłek (macierz pomyłek, ang. confusion matrix)

- Ma na celu zweryfikowanie, które klasy były najczęściej mylone z którymi (w procesie klasyfikacji).
- Wiersze – rzeczywista klasa gestu (obiektu).
- Kolumny – rozpoznana klasa gestu (obiektu).
- Im wyższe wyniki na przekątnej, tym większa dokładność klasyfikatora.
- Można ją wykonać przy każdym rodzaju testów walidacji.



	97	2	1							
		98						1	1	
			100							
				99	1					
				2	96	2				
					2	98				
							95	5		
							1	98		1
							1		99	
				1						99

Ogólna dokładność klasyfikacji: 97,9%