POLITECHNIKA WROCŁAWSKA
WYDZIAŁ ELEKTRONIKI

KIERUNEK: Automatyka i Robotyka
SPECJALNOŚĆ: Technologie informacyjne w systemach automatyki

PRACA DYPLOMOWA MAGISTERSKA

Badanie metod rozpoznawania znaków drogowych z wykorzystaniem wielu źródeł obrazu

Research on traffic sign recognition methods using multiple image source

autor : BARTOSZ LENARTOWICZ

Opiekun pracy:
Dr inż. Bartosz Jabłoński W04/K8

OCENA PRACY:

WROCŁAW 2019

Streszczenie

[here] (streszczenie na sam koniec)

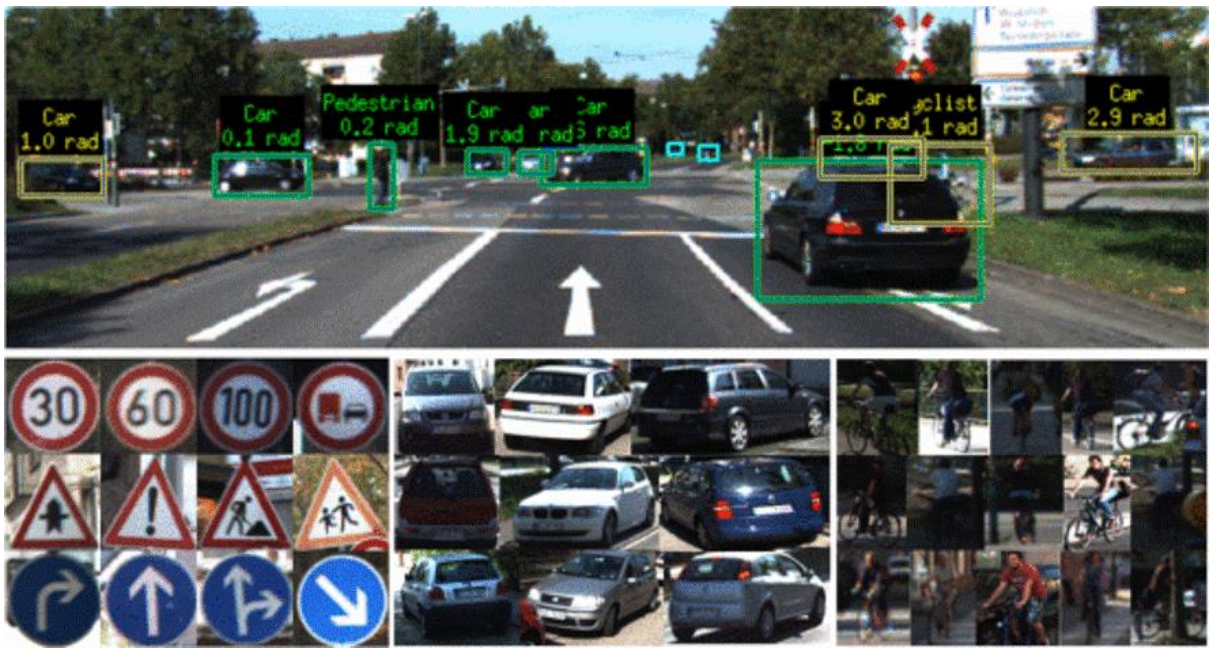
Spis treści

Analiza problemu, cel i zakres pracy

Rozpoznanie znaków drogowych przez algorytmy przetwarzania obrazów jest aspektem niosącym ze sobą wiele korzyści, szczególnie jeżeli analiza następuje w czasie rzeczywistym. Pierwszą korzyścią jaka się nasuwa jest wspomaganie kierowców podczas jazdy ADAS (ang. Advanced Driver Assistance System). Po nieco dłuższym zastanowieniu można wywnioskować, że rozpoznawanie znaków jest niezbędne również dla autonomicznych

pojazdów. Dodatkowo rozpoznanie znaków drogowych może realnie przyczynić się do polepszania bezpieczeństwa na drodze. Przykładowo po rozpoznaniu znaku drogowego można zbadać w jakim stopniu jest on czytelny np. czy nie został zamazany, zakrzywiony. Jeżeli tak się stało można poinformować o tym służbę drogową. Istnieje szereg dziedzin gdzie analiza znaków drogowych może wprowadzić wiele innowacji. Prawdopodobnie dlatego inżynierowie od ponad dwóch dekad starają się sprostać temu zadaniu. Jedna z pierwszych prac nad systemem została przedstawiona w roku 1996 w pracy [1]. W niniejszej pracy zostało przedstawione kilka metod, które starają się sprostać zadaniu rozpoznawania znaków drogowych na obrazie.

Starając się wyjaśnić zakres pracy magisterskiej należy również rozszerzyć kilka terminów, które zostały w niej poruszone. W literaturze problem rozpoznawania znaków drogowych przedstawia się jako problem TSR (ang. Traffic Sign Recognition). Jest on składową większego zagadnienia jakim jest analiza sytuacji w okuł samochodu podczas jazdy, a czasem nawet wewnątrz niego. W takim przypadku obszar wokół samochodu traktowany jest jako scena w, której należy rozpoznawać i analizować obiekty na niej umieszczone. Zostaje to wspomniane, ponieważ niektóre systemy prócz rozpoznania znaków drogowych są w stanie rozpoznawać obiekty takie jak samochody, rowerzystów czy pieszych.



Rysunek 1 Rodzaje obiektów na i wokół drogi

Przykład takiego systemu został przedstawiony w pracy [2]. Zaletą innych systemów analizy sceny TSR jest możliwość śledzenia obiektów np. w celu poinformowania kierowców o kursie kolizyjnym. Śledzenie pozwala również zapobiec wykrycia kilkukrotnie tego samego obiektu. Według autorów pracy [3] aspekt śledzenia jest szczególnie ważny dla systemów, w których pojazd jest kierowany przez kierowcę (a nie przez algorytm). Wielokrotne informowanie kierowcy o znaku może rozproszyć jego uwagę powodując tym samym zagrożenie. Autorzy w swojej pracy wspominają, że traktowanie kierowcy jako integralną część środowiska może polepszyć parametry systemu.

Chcąc zrealizować w skończonym czasie pracę należało wprowadzić pewne ograniczenia. Pierwszym z nich jest problem wykrywania obiektów zawężony do problemu TSR.

Dodatkowo w pracy główny nacisk został położony na wykrywanie znaków pionowych. Warto również na początku zaznaczyć, że skuteczność proponowanego algorytmu była większym kryterium niż jego szybkość.

Główną ideą pracy jest sprawdzenie jaki wpływ na problem TSR ma zastosowanie większej ilości kamer. Obrazy zostały zarejestrowane przez kamery cyfrowe obejmujące obszar przed samochodem [here](oraz części prawego pobocza?). Klatki utrwalone w tym samym czasie zostały wyselekcjonowane i połączone w obraz panoramiczny. Na obrazie panoramicznym nastąpiła detekcja znaków drogowych.

Warto zwrócić też uwagę na fakt, że system, który jest w stanie wykryć kilka różnych typów znaków lecz jest obarczony dużym błędem wykrywalności, w niektórych zastosowaniach może być uważany za lepszy niż system, który może wykryć tylko jeden określony typ znaku, ale robi to bardzo dobrze. Oczywiście możliwa jest również sytuacja odwrotna.

Skupiając się na problemie badawczym czyli wpływie zastosowania kilku kamer, zamiast kamery pojedynczej, należało przyjąć pewne kryteria oceny. Problem oceny został ujęty jako zagadnienie statystyczne. Hipotezą zerową było stwierdzenie, że w przeszukiwanym obszarze nie ma znaków drogowych. Z pozoru nielogiczne założenie wprowadziło kilka udogodnień w rozumowaniu. Tak potraktowane zagadnienie poskutkowało tym, że błędami pierwszego rodzaj było wykrycie znaków drogowych gdzie w rzeczywistości ich nie ma. Błędem drugiego rodzaju zostało ominięcie znaku drogowego na obszarze gdzie w rzeczywistości on występuje.

Ważnym aspektem do testowania algorytmów, zwłaszcza tych do klasyfikacji jest ujednolicona baza danych, ponieważ dopiero na jednolitej bazie danych można określić skuteczność algorytmu. Istnieje kilka publicznie dostępnych zestawów danych znaków drogowych poniżej podano kilka publicznych baz znaków drogowych:

- GTSRB - Niemiecki test TSR,
- KUL Belgium - Zestaw danych znaków drogowych,
- STS - Szwedzki zestaw danych znaków drogowych,
- RUG Traffic Sign - Baza danych obrazów,
- Baza danych Stereopolis.

Najważniejszą bazą danych z pozycji tej pracy jest baza danych GTSRB (ang. German Traffic Sign Recognition Benchmark), ponieważ większość algorytmów przedstawionych w pracy była właśnie testowana na tej bazie danych. Baza posiada zbiór ponad 50 000 obrazów podzielonych na 43 klasy. Obrazy zawierają znaki wypełniające około 70 % obrazu z kawałkami tła. Zostały wyselekcjonowane z sekwencji wideo oraz przycięte. Przykładowe obrazy z tego zbioru pokazane są na obrazie Rysunek 2.

Rysunek 2 Obrazy pochodzące ze zbioru GTSRB

Jak wcześniej zostało wspomniane obrazy z zbioru GTSRB głównie przydatne były do testowania algorytmów klasyfikacji obrazów. Do testowania segmentacji w większości prac tworzone były metody autorskie i testowane na wideo z jazdy.

Metody detekcji obiektów na obrazach panoramicznych

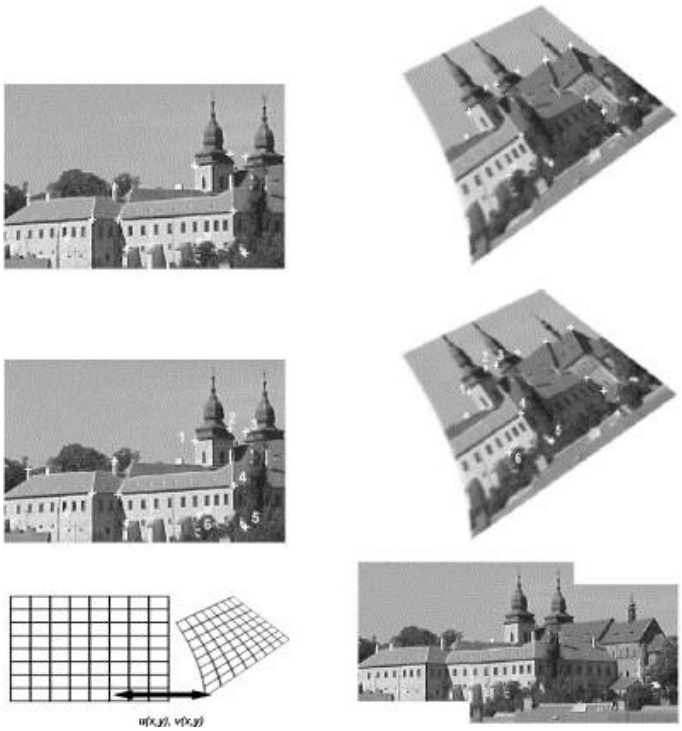
Problem identyfikacji obiektów na obrazach panoramicznych pociąga za sobą szereg zagadnień do zrealizowania. Pierwszym z nich jest stworzenie obrazu panoramicznego z kilku ujęć otoczenia. Po stworzeniu panoramy należy rozpoznać co znajduje się na obrazie. Zagadnienie rozpoznania obrazu w większości przypadków realizowane jest w trzech krokach. Pierwszym krokiem jest przetwarzanie wstępne. Podczas takiego przetwarzania następuje np. akwizycja obrazu. W ogólności przetwarzanie-nisko-poziomowe ma na celu poprawę jakości obrazu poprzez eliminację zakłóceń, poprawę kontrastu, filtrację itp. Kolejny krok nazwany przetwarzaniem średniego poziomu i dotyczy segmentacji obrazu. Podczas segmentacji wydzielone zostają cechy obrazu. Przykładowo znajduje się punkty kluczowe lub obszary, w

których mogą wystąpić obiekty. W pracy etap wstępnego przetwarzania omówiony został wraz z etapem segmentacji. Po segmentacji następuje próba rozpoznania i klasyfikacji obiektu. Bardziej zaawansowane metody pracujące na filmach posiadają również etap śledzenia wykrytych obiektów. Ponieważ każdy etapów przetwarzania obrazów można traktować jako oddzielny problem, poniżej w osobnych podrozdziałach zostały przedstawione metody rozwiązania każdego zagadnienia.

Algorytmy łączenia obrazów w panoramę

Istnieje szereg algorytmów starających się sprostać problemowi dopasowania kilku obrazów do siebie. Istotną cechą poszukiwanego algorytmu powinna być odporność na różnicę w jasności w obrazach wejściowych oraz szumy spowodowane drganiami i rotacją oraz zabrudzeniami typu: błoto, kurz, krople wody. W wielu pracach np. [4] łączenie obrazów w panoramę zostaje podzielone na kilka etapów. W artykule starano się uniwersalną metodę mogącą łączyć obrazy tej samej sceny uzyskane w różnych czasach, z różnych punktów widzenia lub zarejestrowane różnymi czujnikami. Według pracy proponuje się cztery etapy łączenia obrazów przedstawione poniżej oraz przedstawione na rysunku Rysunek 3:

- Wykrywanie cech kluczowych gdzie wykrywa się np. kontury, narożniki, centra grawitacji, punkty kluczowe.
- Dopasowywanie cech gdzie ustalana jest zgodność między cechami wykrytymi w wykrytym obrazie i wykrytymi w obrazie odniesienia. W tym celu używane są różne deskryptory funkcji i podobieństwa oraz relacje przestrzenne między funkcjami.
- Ujednolicenie dwóch modeli. Na tym etapie przekształca się oba obrazy w taki sposób by można było połączyć je w całość za pomocą funkcji odwzorowania. Funkcje odwzorowania umożliwiają wyrównanie obrazów, a nawet zniwelowanie odkształceń. Parametry funkcji odwzorowania są obliczane za pomocą ustalonej zgodności cech.
- Ponowne próbkowanie obrazu i transformacja. Obrazy są transformowane za pomocą funkcji odwzorowania. Wartości obrazu w niecałkowitych współrzędnych są obliczane za pomocą odpowiedniej techniki interpolacji.



Rysunek 3 Przedstawienie 4 kroków tworzenia obrazów

W każdym etapie należy zadecydować, które funkcje będą najskuteczniejsze. Warto pamiętać, by wybrany algorytm wykrywał te same cechy kluczowe na obu obrazach, przy czym każdy z obrazów może być odmiennie zaszumiony. W tej fazie poprzez ekstrakcje cech szczególnych klatki wyłania się punkty kluczowe, niezmiennicze bądź krawędzie obiektów służące do dopasowania obrazów. Po dopasowaniu obrazów następuje fuzja właściwa powodująca scalenie obrazu w panoramę. Warto zaznaczyć, że wyekstraktowane cechy mogą posłużyć w kolejnych etapach do identyfikacji obiektów.

Dziedziną, z której można czerpać inspirację dla fuzji jest medycyna. Przykładem, który można podać jest tomografia lub rezonans magnetyczny, gdzie nie dość, że pojedyncze klatki obrazów są łączone w taki sposób by tworzyć obraz 3-D to jeszcze często na potrzeby dokładniejszej rozeznania łączy się obrazy 3-D z różnych źródeł. Artykuł [5] przedstawia metodę Powella z użyciem metody Brenta na potrzeby łączenia obrazów medycznych.

Poniżej zostały omówione różne algorytmy tworzenia panoramy. Wpierw została przedstawiona idea metod klasycznych, dziś o znaczeniu historycznym. Następnie zaprezentowano metody bazujących na punktach kluczowych, charakterystycznych. Są to metody najpopularniejsze mające największe znaczenie w rzeczywistości.

Metody klasyczne

Metody klasyczne dopasowują obrazy do siebie poprzez wyznaczenie maksimum miary dopasowania pomiędzy cechami wykrytymi na obu obrazach gdzie dopasowanie wykonuje się głównie metodami optymalizacji funkcji. Najprostszym algorytmem do wykrywania cech obrazu jest detektor Canny pozwalający wychwycić krawędzie, zaś najprostszą metodą wyznaczenia miary dopasowania jest suma kwadratów. W istocie obrazy stara się dopasować do siebie starając się zminimalizować błąd średniokwadratowy przez dopasowanie linii na obu obrazach. Wyznaczenie zbieżności funkcji w wielu przypadkach możliwe jest za pomocą

różnych metod np. Gaussa-Newtona, Levenberga-Marquardta. Takie dopasowanie obrazów niesie ze sobą wiele problemów, dlatego starano się znaleźć lepsze metody łączenia obrazów.

W pracy [6] został zaproponowany algorytm do łączenia klatek filmowych tego samego obszaru nagrywanych różnymi kamerami (autorzy testowali algorytm na dwóch kamerach – TV i IR), jednak możliwe jest przystosowanie algorytmu do tworzenia panoramy. Algorytm przeskalowuje obrazy do jednej wielkości, a następnie poddaje je filtrowi krawędziowemu. Autorzy pracy nie podają jaki dokładnie filtr był zastosowany. Następnie jeden obraz zostaje podzielony na pionowe wycinki i każdy wycinek stara się dopasować do obrazu drugiego. Najlepsze dopasowanie każdego odcinka wyznacza się za pomocą sumy najmniejszych kwadratów lub sumy modułów różnic. Następnie za pomocą jednej z metod: średniej, mediany lub dominanty wybiera się najlepsze dopasowanie ogólne. Na podstawie dopasowania ogólnego zostaje wyznaczona translacja pozioma obrazów. Warto zaznaczyć, że dominanta uzyskała najlepsze wyniki dopasowania. Gdy obrazy zostają przesunięte o wyznaczoną wartość i proceder powtarza się w pionie. Algorytm jest słabo odporny na rotację lecz jego największą zaletą jest szybkość.

Można również wspomnieć o metodzie przedstawionej w artykule [7]. W skrócie metoda polega podzieleniu obu obrazów na niewielkie obszary i dopasowaniu obrazów do siebie za pomocą szybkich, mało dokładnych metod z wykorzystaniem miary statystycznej korelacji. Następnie używając metod bardziej złożonych obliczeniowo ale dokładniejszych łączy się segmenty w całość i znajduje najlepsze globalne dopasowanie. Autorzy algorytmu w opisie przedstawiają, że algorytm wykorzystuje metodę quasi-Newtona, z piramidą obszarów. Takie łączenie obrazów pozwala na większą eliminację błędów.

Metody bazujące na punktach charakterystycznych

Metody łączenia obrazów, które posiadają większą odporność na zakłócenia bazują na ekstrakcji cech. W literaturze punkty charakterystyczne nazywane są również punktami, kluczowych bądź kontrolnymi CP (ang. Control Point). Cechy takie rozumiane są jako specyficzne konfiguracje pikseli układające się w struktury. Przykładami takich konfiguracji mogą być: zakończenia linii, krawędzie lub kąty. Wykrywanie struktur zazwyczaj następuje za pomocą filtracji obrazu. Fragmenty obrazu, które nie zmieniają się podczas przekształceń obrazu nazywane są punktami kluczowymi. Ważnym elementem punktów kluczowych jest to ich skalo-niezmienniczość.

Detektor Harris'a

Jako punkty charakterystyczne można wykorzystać narożniki wykryte na obrazach. Detektor Harrisa, który jest ulepszoną wersją detektora Moraveca z powodzeniem wykrywa narożniki. Koncepcja detektora Moraveca polega na przeszukiwaniu obrazu z wykorzystaniem okna przeszukiwania ROI (ang. Region of Interest). Gdy podczas przeszukiwania w jednym kierunku zostanie zauważona duża zmiana w jasności pikseli obszar zostaje zakwalifikowany jako krawędź. Jeżeli zmiana zostanie również zauważona w kierunku prostopadłym to obszar kwalifikuje się jako narożnik. Zmiany intensywności sprawdza się co 45° co jest istotną wadą. W detektorze Moraveca wadami są również zaszumiona odpowiedź z uwagi na binarną funkcję okna oraz minimum jako kryterium. Chcąc poprawić algorytm Chris Harris w 1988 roku zaproponował ulepszoną wersję algorytmu z powodzeniem stosowaną do dnia dzisiejszego. W detektorze Harrisa wykorzystywana jest macierz autokorelacji w postaci:

$$M = \begin{bmatrix} (\frac{\partial I}{\partial x})^2 e^{\frac{-(x^2+y^2)}{2\sigma^2}} & (\frac{\partial I}{\partial x})^2 (\frac{\partial I}{\partial y})^2 e^{\frac{-(x^2+y^2)}{2\sigma^2}} \\ (\frac{\partial I}{\partial x})^2 (\frac{\partial I}{\partial y})^2 e^{\frac{-(x^2+y^2)}{2\sigma^2}} & (\frac{\partial I}{\partial y})^2 e^{\frac{-(x^2+y^2)}{2\sigma^2}} \end{bmatrix}$$

Równanie 1 Macierz autokorelacji detektora Harrisa

gdzie I jest funkcją intensywności, a σ jest odchyleniem standardowym funkcji Gaussa. Jeżeli różnica między wyznacznikiem macierzy M, a kwadratem jej śladu przemnożonym przez stałą k ($P(x,y) = \det(M) - k(trace(M))^2$) będzie mniejsza od zera $P(x,y) < 0$ to punkt $P(x,y)$ jest uważany za krawędź. Jeżeli zaś $P(x,y) > 0$ to punkt $P(x,y)$ uważany jest za narożnik. Jeżeli punkt jest bliski zera $P(x,y) \sim 0$ to uważa się, że obszar nie posiada znaczących zmian. Dodatkowo dla większej dokładności algorytmu wykrywania narożników wprowadza się próg $P(x,y) > T$. Zaleca się również wybrania punktów z lokalnym maksimum. Detektor Harrisa wykrywa narożniki z wiele większą dokładnością niż detektor Moraveca oraz sprawdza narożniki pod każdym kątem. Algorytm może z powodzeniem zostać wykorzystywany do tworzenia panoram.

GFTT

Algorytm GFTT (ang. - Good Features To Track) jest kolejnym ulepszeniem jakie wprowadzili Jianbo Shi oraz Carlo Tomasi w 1994 roku dla detektora Harrisa. Na potrzeby algorytmy wymaga się by obraz wejściowy był czarno biały. Zmianie w odniesieniu do oryginalnego algorytmu polegała na zastosowaniu innego wzoru wyliczającego funkcje oceniającą. Zamiast jak w oryginale $P(x,y) = \det(M) - k(trace(M))^2$ w przypadku GFTT funkcja ta ma postać

$$P(x,y) = \min(\left(\frac{\partial I}{\partial x}\right)^2 e^{\frac{-(x^2+y^2)}{2\sigma^2}}, \quad \left(\frac{\partial I}{\partial y}\right)^2 e^{\frac{-(x^2+y^2)}{2\sigma^2}})$$

Równanie 2 Funkcja oceniająca dla algorytmu GFTT

Dzięki swojej modyfikacji algorytm znajduje najbardziej użyteczne wierzchołki obrazu. Każdy wykryty wierzchołek sprawdzana się czy przekroczył pewien próg zmiany jasności i jeżeli tego nie zrobił zostaje automatycznie odrzucony. Dodatkowo określa się minimalny dystans pomiędzy wierzchołkami. Kryteria pozwalają wyłonić najlepiej procentujące wierzchołki, czyli intersujące punkty kluczowe.

FAST

Detektor FAST (ang. Features from Accelerated Segment Test) w swoim kryterium porównuje jasność piksela z jasnością pikseli oddalonych o ustalony promień. W przypadku gdy jasność większości pikseli będzie się różnić od piksela centralnego o określoną wartość to punkt kwalifikowany jest jako narożnik. Przykładowo dla promienia równego 3 piksele jeżeli 12 z 16 pikseli będą jaśniejsze niż piksel centralny pomniejszony o pewną wartość to algorytm wykrył narożnik. Na potrzeby algorytmu promień powinien być dostatecznie mały (od 2 do 6 pikseli) oraz wartość progu kwalifikacji i ilości pikseli odpowiednio przeskalowana do promienia. Algorytm FAST osiąga przeciętne wyniki w wykrywaniu punktów kluczowych lecz jego zaletą jest szybkość.

SIFT

Algorytmem który wykorzystuje punkty charakterystyczne na którego należało by zwrócić szczególną uwagę jest zaproponowany przez David Lowe algorytm SIFT (ang. Scale Invariant

Feature Transform) [8]. Jest to jeden z bardziej popularnych algorytmów dlatego jego działanie zostało przybliżone w tym akapicie. Algorytm SIFT realizowany jest w czterech krokach. Kroki zostały wpierw przedstawione, a następnie szczegółowo opisane. Pierwszy krok nazwany „scale space extrema detection" wykrywa punkty ekstremalne na dwóch obrazach. W kroku drugim „accurate keypoint location" następuje dokładna lokalizacja punktów charakterystycznych. Następnie przypisuje się orientację wykrytym punktom w przestrzeni. Krok nazwany jest „keypoint orientation assignment". Czwarty etap zawiera tworzenie deskryptorów dla punktów charakterystycznych. Szczegółowe omawianie algorytmu należy zacząć od procesu skalowania obrazu wejściowego do różnych wielkości. Dla każdego obrazu w skali zostaje użyty filtr Laplace'a, który pozwala uzyskać kontury obiektów D_n , obliczany za pomocą odjęcia dwóch obrazów rozmytych filtrem Gaussa z różnymi parametrem σ. Zazwyczaj większe rozmycie obrazu tworzy się poprzez zwiększenie potęgi do której zostanie podniesiona stała k. Dla ułatwienia zrozumienia poniżej opisano wzór

$$D_n = \frac{1}{2\pi(\sigma k^{n+1})^2} e^{\frac{-(x^2+y^2)}{2k^2\sigma^2}} \cdot P(x,y) - \frac{1}{2\pi(\sigma k^{n})^2} e^{\frac{-(x^2+y^2)}{2k^2\sigma^2}} \cdot P(x,y)$$

Zazwyczaj w za parametr odchylenia standardowego przyjmuje się $\sigma = 1{,}6$. Stała $k = \sqrt{2}$ podnoszona do kolejnych potęg. Następnie przez binaryzację wykrywa się lokalne maksima i minima. Taka operacja powoduje wykrycie ogromnej ilości punktów ekstremalnych dlatego należy zastosować dwa kryterium, które odfiltrują najlepsze możliwe punkty charakterystyczne. Pierwsze kryterium odrzuca płytkie minima bądź niewielkie maksima za pomocą rozwinięcia funkcji w szereg Taylora. Dzięki temu odrzucone punkty, które powstały na obszarze np. nieba. Drugie kryterium sprawdza czy punkt nie leży na odcinku za pomocą detektora Harris'a. Po tym etapie wyłonione zostają punkty niezmiennicze. Algorytm przypisuje punktom orientację co powoduje niezmienniczość również względem orientacji. Można więc ustawić obrazy tak by odpowiadające sobie punkty charakterystyczne na każdym obrazie wskazywały jednakowy kierunek. Realizowane jest to poprzez wyznaczenie gradientu w punkcie charakterystycznym. Następnie wyznacza się gradienty w małym otoczeniu punktu charakterystycznego i z odpowiednio wyważonych gradientów tworzy się histogram. Z histogramu powstaje deskryptor który pozwala wyznaczyć orientację obszaru przetwarzanego w końcowym etapie. Etap końcowy polega na tworzeniu ostatecznych deskryptorów. Najczęściej przetwarzaniu podlega rozmyty wejściowy obraz. Etap przypomina ten poprzedni lecz otoczenie punktu charakterystycznego jest dużo większe i podzielone na cztery obszary. W każdym obszarze wyznacza się osobne deskryptory co kończy algorytm. Uzyskuje się w ten sposób punkty charakterystyczne wraz z dokładną orientacją.

Łączenie obrazów

Chcąc zakończyć tematykę tworzenia panoramy należało by wspomnieć w jaki sposób za pomocą metod klasycznych oraz tych bazujących na punktach kluczowych zostaje utworzony jednolity obraz panoramiczny. Nie trudno zauważyć, że w każdej z metod wykryte cechy pozwalają dopasować obrazy do siebie. Następnie nakłada się jeden obraz na drugi w taki sposób by występowała największa liczba pasujących elementów np. wykrytych linii, krawędzi, narożników czy deskryptorów. Obrazy zostają scalone w jeden tzn. dokonuje się ich fuzji. Zazwyczaj scalenie polega na przysłonięciu części obrazu drugim obrazem. Istnieją również prace, które starają się wspólny obszar zapełnić informacjami z obu obrazów lecz takie rozwiązanie może spowodować pogorszenie się jakości obrazu.

Segmentacja obrazów

Segmentacja obrazów polega na znalezieniu obszaru, w którym występuje poszukiwany obiekt. W przypadku systemu TSR segmentacja obrazów polega na znalezieniu na obrazie znaku drogowego. Wykrywanie obiektów na obrazach może się odbywać na wiele sposobów. Kilka metod wykrywania i rozpoznawania obiektu zostały przedstawione poniżej. Wpierw zostało omówione podejście na podstawie koloru, a następnie kształtu. W kolejnym akapicie omówiono podejście odnoszące się do tekstury obiektu po czym zostały omówione metody hybrydowe łączące wyżej przedstawione podejścia. Na sam koniec zostało wspomnianych kilka innych metod.

[here](usunąć cały poniższy opis?)

By takie wykrywanie mogło mieć miejsce należy wyznaczyć wzorzec znaku zbiór cech tworzący jakościowy i ilościowy opis obiektu (znaku drogowego). Zwykle wzorzec opisywany jest za pomocą wektora cech: $x = [x_1, x_2, \dots, x_n]$*. Dla ułatwienia opisu obiektów zostały również wprowadzone klasy wzorców, zawierające wzorce z podobnymi wektorami cech. Klasę wzorców oznacza się za pomocą* $\omega_1, \omega_2, \dots, \omega_M$ *(M - numer klasy). Rozpoznanie wzorców nazwane klasyfikacją polega na przyporządkowaniu wzorców do ich klas. Wyznaczenie wzorców powinno się opierać o starannie dobrane cechy. Cechy takie powinny mieć specyficzne własności. Ważne jest by wybrane cechy przypisane do obiektu przyjmowały różne wartości dla różnych klas obiektów. Ważnym atrybutem każdej cechy powinna być jej niezawodność. Niezawodność powinna polegać na przyjmowaniu podobnych wartości dla wszystkich obiektów danej klasy. Dodatkowo cechy wybrane wzorca powinny być nieskorelowane ze sobą. Skorelowanie można opisać za pomocą współczynnika korelacji:* $\sigma_{x,y} = \frac{\frac{1}{P}\sum_{i=1}^{P}(x_i-\mu_x)(y_i-\mu_y)}{\sigma_x\sigma_y}$ *gdzie P oznacza liczbę klasyfikowanych obiektów,* μ *wartości średnie, a* σ *odchylenie standardowe zbioru cech. Jeżeli współczynnik korelacji* $\sigma_{x,y} = 1\ (-1)$ *To cechy x i y uważa się za silnie skorelowane i jedną z nich można odrzucić. Ważne jest by dla obiektu nie przyjmować zbyt dużej ilości cech, ponieważ złożoność obliczeniowa detektorów obiektów rośnie wraz z ilością cech.*

Wykrywanie na podstawie koloru

Wykrywanie obiektów na podstawie koloru najczęściej stosuje się w przypadku kolor obiektu jest nieskomplikowany i dobrze określony. Jak można się domyślić by wykrywanie na podstawie koloru mogło się odbyć obiekt, który chcemy wykryć powinien posiadać z góry określone kolory. Znaki drogowe posiadają taką własność oraz projektowane są w taki sposób by ich kolor znacznie odróżniał się od tła. Najpopularniejszą metodą wykrywania obiektów na podstawie koloru jest progowanie obrazu w określonej barwie. Zaletą progowania jest jego szybkość. Ważnym jest by kolor znacząco odróżniał się od tła. Znaczna część przedstawionych prac np. [9] [10] wykorzystuje progowanie obrazu w kolorze czerwonym by wstępnie wyszukać regiony gdzie mogą znajdować się znaki. Określenie koloru znaku może być zrealizowane za pomocą kilku sposobów lecz zazwyczaj wykonuje się to ręcznie przez pobranie próbek z kilkunastu wzorcowych obiektów. Następnie wyznaczony zostaje średni kolor i możliwy zakres odchylenia od wzorca. Warto zauważyć, że progowanie jest mało odporne na wszelkiego typu zakłócenia, ponieważ kolor obiektu w dużej mierze zależy od oświetlenia. By zaradzić problemowi zmiany oświetlenia w zależności od warunków część algorytmów wykorzystuje możliwość zmiany przestrzeni barw.

Zmiana przestrzeni barw

Przestrzeń barw RGB (ang. Red, Green, Blue) jest bardzo zmienna w zależności od warunków panujących na drodze. Pogoda, pora dnia lub mnogość różnokolorowych źródeł światła mogą spowodować zmianę odbieranych przez kamerę kolorów. Część algorytmów przed progowaniem wstępnie obrabia obraz stosując techniki tj. rozciągnięcie histogramu. Metody te są jednak mało skuteczne dlatego część naukowców badających temat TSR postanowiło przejść z przestrzeni barw RGB na HSI (ang. Hue, Saturation, Intensity) lub HSV (ang. Hue Saturation Value) [11], [12], [13]. Takie modele barw pozwalają w pewnym stopniu

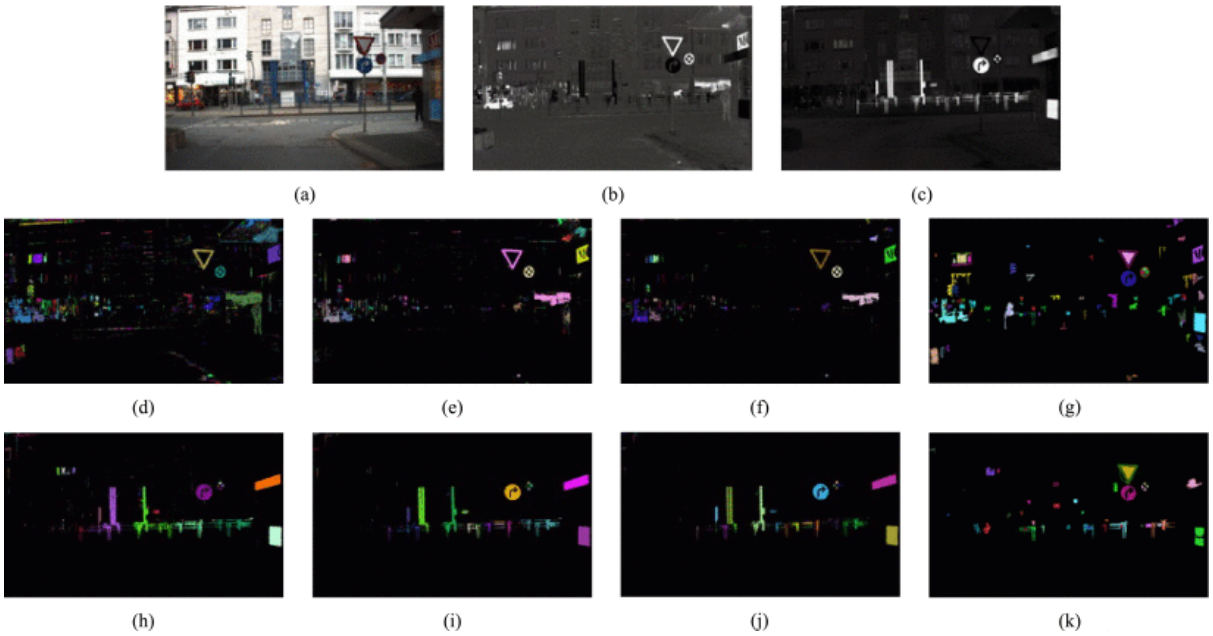
zniwelować wahania spowodowane zaburzeniami oświetleniowymi. Niestety przestrzenie HSI i HSV nie są wolne od wad, przykładowo nie uwzględniają temperatury barwy. Przestrzeń barw LCH (ang. Lightness, Chroma, Hue) uzyskana za pomocą modelu CIECAM97 pozwala uwzględnić temperaturę barw. Warto wspomnieć również o przestrzeni YUV, która posiada trzy kanały. Jeden kanał Y dostarczający luminację czyli jasność obrazu i dwa kanały U i V kodującą chrominancję czyli barwę, ponieważ algorytm bazujący na splotowej sieci neuronowej [14] wykorzystuje właśnie przejście na taką przestrzeń barw. By opis był pełny należy zaznaczyć, że część autorów w swoich pracach stwierdza, że zmiana przestrzeni barw jest zbędna, ponieważ unormowana przestrzeń RGB w przeciętnych warunkach jest wystarczająca, a koszt obliczeniowy związany ze zmianą przestrzeni barw jest zbyt duży.

AdaBoost

Warto dodać, że w pracy [10] model został oparty na kolorach ale nie na samym progowaniu. Algorytm używa zagnieżdżonej kaskady klasyfikatora Real AdaBoost, którą szkoli się za pomocą cech LRP (ang. Local Rank Pattern). Do obliczenia cech LRP wykorzystuje się siedem typów obrazów, gdzie każdy obraz tworzy się przez odjęcie jednego typu koloru (np. czerwonego, niebieskiego zielonego itp.) W ogólności algorytm AdaBoost jest bardzo popularny i został wykorzystany w ogromnej liczbie prac. AdaBoost to oparty na boostingu algorytm zaprezentowany 1996 roku przez Yoav Freund i Robert Schapire [15]. Główna idea polega na stworzeniu z wielu słabych klasyfikatorów jednego silnego klasyfikator.

Model prawdopodobieństwa barwy

Algorytm korzystający z modelu prawdopodobieństwa barw do wykrywania znaków drogowych został przedstawiony w pracy [16]. Algorytm tworzy mapę prawdopodobieństwa występowania znaku drogowego na podstawie poszukiwania kolorów znaków drogowych w obrazie. Następnie w regionach gdzie jest najwięcej poszukiwanych kolorów algorytm stara się wykryć znak. Dodatkowo poszukiwanie koloru następuje w przestrzeni kolorów Otha, ponieważ taka przestrzeń lepiej sprawdziła się w eksperymentach prowadzonych dla pracy. Skanowanie w przestrzeni Otha zostało przedstawione na Rysunek 4.



Rysunek 4 Tworzenie mapy prawdopodobieństwa w przestrzeni Otha

Na wyżej pokazanym Rysunek 4 obraz (a) jest wejściowym obrazem w kolorze. Obrazy Rysunek 4(b) i (c) są mapami prawdopodobieństwa uzyskanymi odpowiednio dla kolorów czerwonego i niebieskiego. Jak można zauważyć, czerwone i niebieskie piksele na oryginalnym występują w dużej ilości i mają spore natężenie. Zastosowanie map prawdopodobieństwa ułatwia wykrywanie znaków przez zwiększenie kontrastu między znakami drogowymi, a tłem. Aby algorytm pracował w czasie rzeczywistym, obliczenia zostały przyspieszone za pomocą wprowadzenia tabeli wyszukiwania LUT (ang. Look Up Table). Podczas wykrywania obliczony zostaje indeks każdego piksela według wartość RGB, a następnie znajduje odpowiadające mu prawdopodobieństwo w tabeli LUT. Autorzy algorytmu twierdzą, że po wprowadzeniu tabeli czas obliczenia map prawdopodobieństwa dla obrazu w rozdzielczości 1360×800 może został zredukowany z kilku minut do około 30 ms komputerze czterordzeniowym procesorem Intel 3,4 GHz, 4 G RAM. Dodatkowo autorzy wykorzystali detektor regionów MSER (ang. Maximally Stable Extremal Regions) aby na mapach prawdopodobieństwa zlokalizować miejsca występowania znaków. Działanie detektora MSER przedstawione zostało na obrazach od (d) do (f) dla progowania w kolorze czerwonym i od (h) do (j), dla koloru niebieskiego. Obrazy (g) i (k) sumę obrazów dostarczonych przez detektor. Na obrazach (g) i (k) następuje wykrycie obszarów ze znakami przez ztworzenie map prawdopodobieństwa. Po wykryciu regionów ze znakami następuje klasyfikacja regionów do poszczególnych podklas poprzez maszynę wektorów nośnych przeszkolonych dla cech HOG z uwzględnieniem koloru. Algorytm HOG został przedstawiony w podrozdziale opisującym wykrywanie znaków na podstawnie tekstury, dlatego w tym miejscu nie został opisany.

Wykrywanie na podstawie kształtu

Powszechnym podejściem w wykrywaniu obiektów na obrazach jest wykrywanie poprzez kształt. Podobnie jak w przypadku koloru tak i w przypadku wykrywania na podstawie kształtu ważną zaletą obiektu musi być konkretnie określony kształt. Jak dla znaków drogowych taki kryterium jest spełnione. Znaki informacyjne są kwadratowe, nakazu lub zakazu okrągłe, a ostrzegawcze trójkątne. Wykrywanie na podstawie kształtu nie jest również wolne od wad. Percepcja kształtów różni się od kąta obserwowania. Innym problemem jest fakt iż podczas obserwacji znaki mogą być częściowo przysłonięte lub na ich część może padać cień, co zmienia ich wygląd dla większości algorytmów. Ważnym jest by zastosowany detektor radził sobie z takimi problemami. Wykrywanie figur geometrycznych na obrazie poprzez kształt może być zrealizowane na kilka sposobów. Te ważniejsze dla pracy zostały przedstawione poniżej.

Wykrywanie krawędzi

Wykrywanie krawędzi jest popularną metodą uwidaczniania obiektów na obrazie. W poprzednim rozdziale opisywano wykrycie krawędzi na potrzeby łączenia obrazów w panoramę. W tym miejscu należy objaśnić, że krawędź identyfikowana jest jako przejście z obszaru ciemniejszego do jaśniejszego bądź na odwrót. Można stwierdzić, że jest to granica pomiędzy dwoma obszarami o różnych jasnościach. Takie podejście wymaga ustalenia konkretnego progu zmiany jasności powodującego wykrycie. Część metod bazuje na operatorach gradientowych ustalając próg lokalny, zamiast globalnego. Takie metody wykorzystują zmiany pierwszej lub drugiej pochodnej obrazu w skali szarości. Powszechnym rozwiązaniem wykrywania krawędzi jest detektor Canny który został omówiony poniżej lecz istnieją również inne metody. Przykładem metody odmiennej od Canny jest Operator Robertsa

oraz Operatora Sobela. Te operatory sprawdzają różnicę miedzy sąsiednimi pikselami w obrazie przez splot pewnej macierzy z obrazem. Algorytmy bazujące na takich operatorach przeznaczone do problemu TSR przedstawione zostały np. w pracy [17].

Detektor Canny

Jak wcześniej zostało wspomniane jedną z popularniejszych metod wykrywania krawędzi jest metoda Canny zaprezentowana w Johna F. Canny w 1986. Metoda przedstawia się następująco. Pierw następuje filtracja obrazu za pomocą filtru Gaussa. Powoduje to wstępnie odfiltrowanie zakłóceń. Następnie za pomocą np. operatora Sobela wykryty zostaje gradient zmian poziomych G_x i pionowych G_y każdego punktu na obrazie. W kolejnym kroku wyznacza się długość $G = \sqrt{G_x^2 + G_y^2}$ i kąt detekcji krawędzi $\theta = \arctan(\frac{G_y}{G_x})$. Niekiedy kąt zaokrągla się do wartości liczonych co 45˚. Piksele, które nie łączą się z żadną pobliską krawędzią zostają odrzucone. Piksele umieszone blisko zakończeń wykrytych krawędzi zostają połączone w jedną krawędź. Na tak przetworzonym obrazie należy odszukać obiekty, które nas interesują. Dla problemu TSR są to: trójkąty, kwadraty i okręgi. Detektor Canny dzięki swoim właściwością doskonale nadaje się do uwidaczniania kształtów obiektów dlatego jest to algorytm po który najczęściej się sięga gdy wymagane jest wykrycie konturów.

Wykrywanie rogów

Wykrywanie rogów może być przydatne zwłaszcza dla znaków ostrzegawczych (trójkąty) i informacyjnych (kwadraty). Wykrywanie narożników może odbywać się np. za pomocą detektora Harrisa opisanego przy okazji metod bazujących na punktach charakterystycznych lub innych metod z opisanych w poprzednim etapie. Jeżeli nastąpi wykrycie rogów w konkretnej konfiguracji adekwatnej dla znaku można domniemywać, że w tym miejscu wystąpił znak i przystąpić do klasyfikacji znaku.

Transformata Hougha

Algorytmem na który należało by również zwrócić uwagę jest filtr (transformata) Hougha, który pozwala odnajdywać regularne kształty na obrazie. Został on zaprezentowany w 1962 roku jako metoda wykrywania linii prostych ale udało się go zastosować dla wykrywania innych regularnych kształtów takich jak okręgi, kwadraty czy trójkąty [18]. W 1993 roku Anagnou przestawił udoskonaloną metodę pozwalającą uzyskanie większej rozdzielczości kątowej oraz radialnej znajdowanej linii. Omówienie metody zostało zrealizowane na przykładzie wykrywania kształtu kwadratowego. W tej metodzie każda prosta opisana jest równaniem $\rho = x\cos\theta + y\sin\theta$. Początkowo zostaje zaimplementowana tablica (komórki nazywa się akumulatorami) z wartościami zerowymi. Następnie każdy punkt obrazu przekształcany jest w dyskretną krzywą sinusoidalną w przestrzeni OρΘ. Oblicza się wartości dla Θ=(0,360˚). Obliczenie wartości ρ uzyskuje się przez dodanie jedynek w odpowiednie pola tablicy. Po wykonaniu inkrementacji dla wszystkich punktów komórki z największą wartością ρ. Położenie komórki określają parametry prostej zaś wielkość ρ określa długość odcinka. Jeżeli wykryte proste mają podobną długość i przecinają się pod kontem prostym to można uznać, że kształt jest kwadratowy. Transformacja Hougha odnosi sukcesy w wykrywaniu linii czy okręgów lecz do wykrycia innych kształtów jak np. trójkąty wymaga skomplikowanych obliczeń i jest obciążająca pamięciowo. W pracy [19] użyto transformaty Hougha do rozróżnienia typu znaku. Praca [19] została bliżej przedstawiona w rozdziale klasyfikacji znaków i metodzie SVM.

Wykrywanie na podstawie tekstury

Z związku z tym, że podczas wykrywania obiektów na podstawie koloru i kształtu uzyskanie dobrych wyników podczas niekorzystnych warunków może być problematyczne, zaczęto szukać innych rozwiązań mogących poprawić skuteczność wykrywania. Zaczynając omawianie wykrywania na podstawie tekstury należało by wyjaśnić, że tekstura to graficzny odpowiednik faktury. Faktura obiektów w rzeczywistości bardzo wpływa na postrzeganie ich przez człowieka dlatego wiele grup zajmujących się odnajdywaniem obiektów na obrazie podejmowało próby skonstruowania algorytmu bazujących na teksturze. Kilka takich metod zostało przedstawione poniżej.

HOG

HOG (ang. Histograms of Oriented Gradients) - histogram gradientów zorientowanych jest deskryptorem obrazu pozwalającym na znalezienie kształtu i wyłonienie obiektu [20]. Idea algorytmu polega na podzieleniu obrazu przekonwertowanego do skali szarości na małe fragmenty (np.8x8 px.) i obliczeniu dla każdego piksela w takim fragmencie różnicy jasności pomiędzy pikselami sąsiadującymi (gradientu). Następnie dla każdego z fragmentu wyznacza się histogram gradientu. W przeciwieństwie do wcześniej przedstawionego algorytmu SIFT, HOG oblicza deskryptory w równomiernie oddalonych fragmentach obrazu. Zastosowanie lokalnej normalizacji kontrastu w nakładających się na siebie regionach poprawia skuteczność wykrywania obiektów [21]. Zaproponowany w pracy [22] wariant HOG (HOGv) posiada dwie modyfikacje poprawiające wykrywanie obiektów. Pierwszą z nich jest uwzględnienie zarówno wrażliwych jak i niewrażliwych na kontrast orientacji gradientów, w taki sposób, że bardziej szczegółowe lokalne informacje o znakach mogą być włączone do zgromadzonych histogramów. Drugą modyfikacją jest to, że każdą komórkę normalizuje się odpowiednio z czterema sąsiednimi blokami. Znormalizowane histogramy komórki są wymiarowo zmniejszane za pomocą strategii podstawowej analizy komponentów (PCA). Takie działanie ma na celu usunięcie nadmiarowych informacji.

LBP

LPB (ang. local binary patterns) zostało opisane w 1994 roku [23] jako algorytm wykrywania obiektów na obrazi. Algorytm odniósł duży sukces w wykrywaniu twarzy dlatego starano się go zaimplementować do wykrywania innych obiektów. Jego idea jest podobna do algorytmu HOG, ponieważ na początku obraz zostaje podzielony na mniejsze obszary. Następnie porównuje się, każdy piksel z wszystkimi pikselami sąsiadującymi w ustalonej kolejności. W zależności czy wartość piksela porównywanego jest większa lub równa czy mniejsza od piksela centralnego w osobnej tablicy zostaje wpisana odpowiednio wartość 1 lub 0. Takie rozwiązanie niesie ze sobą zaletę, że gdy zostaje zmieniona globalna jasność obrazu, wartości tablic [0,1,(…)] się nie zmienią i skuteczność algorytmu nie spadnie. Następnie podobnie jak w algorytmie HOG w fragmentach zostaje utworzony histogram, a następnie zostaje ustalony gradient całego okna. Po wyznaczeniu gradientów w każdym oknie następuje połączenie podobnych gradientów mogących tworzyć jeden obiekt. Jeżeli odnalezione pole wyznaczone przez gradienty jest podobne do jednego z obiektów przechowywanych w bazie algorytm kończy działanie. LBP dla wykrywania znaków drogowych zaproponowano w pracy [19]. Rozwiązanie w zbogacone jest o ustalenie na początku działania potencjalnych obszarów poszukiwań znaków za pomocą okna przesuwanego o niewielkich rozmiarach. Etap nazywa się filtrowaniem zgrubnym i używa algorytmu LDA. Następnie podczas dokładnego filtrowania weryfikuje się okna odnalezione w etapie poprzednim za pomocą metody NMS

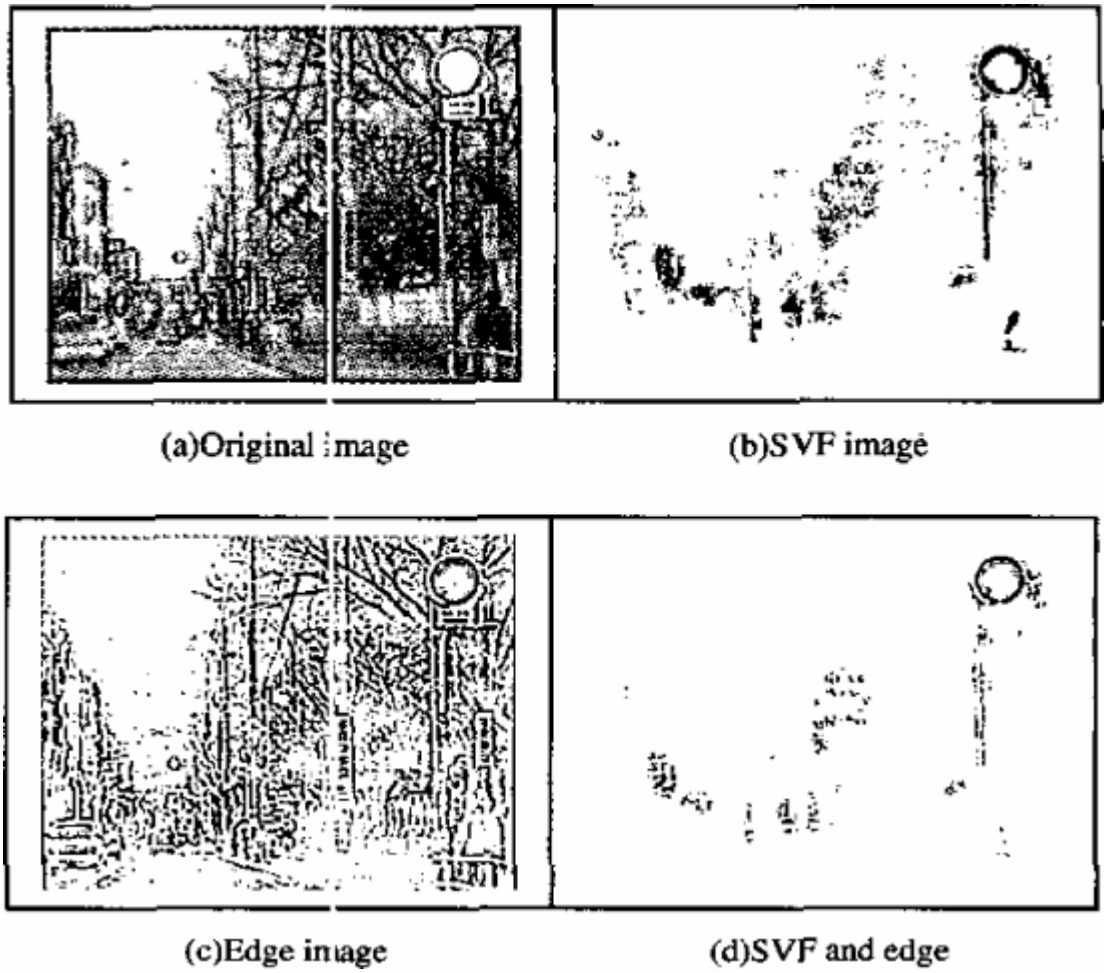
(ang. Non-Maximal Suppression) wykonywane jest filtrowanie okien mogących odnosić się do tego samego obszaru. Po odnalezieniu obszarów gdzie możliwe jest wystąpienie znaków drogowych ich klasyfikacja wykonywana jest za pomocą maszyny wektorów nośnych. Maszyna ta została opisana w następnym rozdziale.

Podejście hybrydowe

Podejście hybrydowe w ogólności polega na połączeniu metod opartych na kolorze, kształcie i fakturze. W większości prac na początkowym etapie następuje progowanie w określonych kolorach co zawęża obszar poszukiwań, a następnie wykorzystuje się podejścia oparte na kształcie i fakturze w celu poprawy wydajności wykrywania.

Filtr krawędziowy z SVF

Przykładem metody hybrydowej stworzonej do segmentacji obiektów na obrazie jest metoda bazująca na filtrze krawędziowym z SVF zaproponowana np. w artykule [12]. W podanej pracy wpierw następuje wyodrębnienie kształtów przez zastosowanie filtra krawędziowego bazującego na zmianie jasności. Niestety takie działanie wyodrębnia wszystkie obiekty sceny. Należy więc odróżnić znaki drogowe od tła. Bazując tylko na filtrze krawędziowym największym problemem są obiekty, których kształt zbliżony jest do kształtu znaków. By poradzić sobie z tą przeszkodą autorzy zaproponowali autorski filtr SVF (ang. Simple Vector Filter) uwzględniający również kolor znalezionego obiektu. Zaletą proponowanego filtru jest jego szybkość. Dokładny opis przebiegu algorytmu jest bezcelowy ponieważ można go doczytać w podanej pracy, lecz skuteczność SVF w odfiltrowaniu znaków drogowych od reszty obiektów przedstawiono na Rysunek 5. By prawdzić czy w wykrytym regionie istnieje znak zastosowano algorytm genetyczny opisany poniżej.



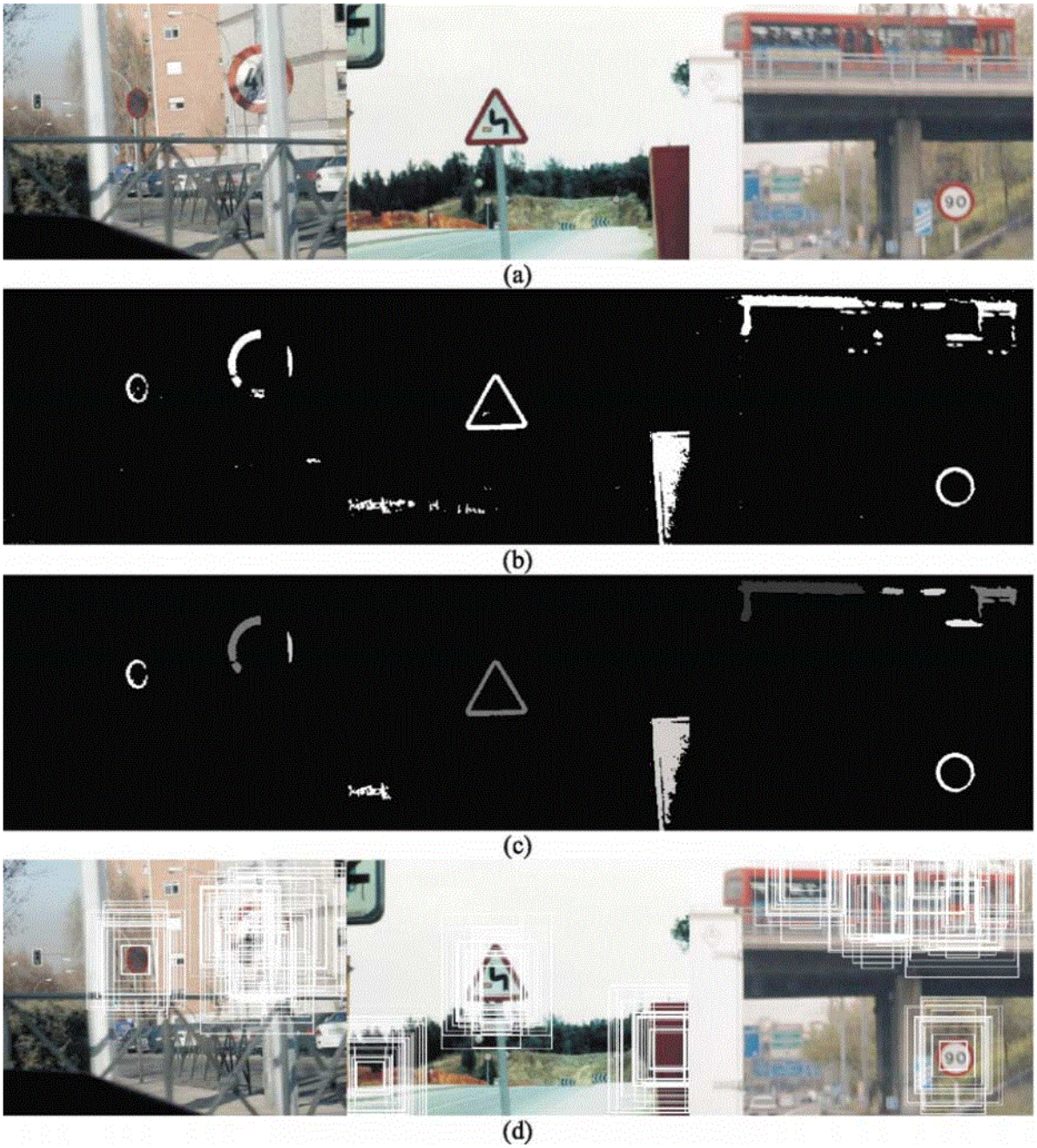
Rysunek 5 Wykrywanie znaków z zastosowaniem filtra SVF

GA

Algorytmy genetyczne są popularnym podejściem w segmentacji i klasyfikacji znaków drogowych. GA (ang. Genetic Algorithm) stara się znaleźć lepsze rozwiązanie poprzez krzyżowanie i mutacje znalezionych rozwiązań w każdej kolejnej iteracji. Początkowe wartości rozwiązań są losowane ze zbioru możliwych rozwiązań, a algorytm kończy działanie gdy wyszukane rozwiązanie jest wystarczająco skuteczne. Rozwiązanie problemu klasyfikacji znaków drogowych z pomocą algorytmu genetycznego przedstawione zostało w wcześniej opisywanej pracy [12]. Algorytm skupia się na wgrywaniu znaków okrągłych na obrazach dostarczonych przez filtr krawędziowy i SVF. Oczywiście znaki okrągłe można wyszukać na obrazach przez znalezienie okręgów. W TSR dużym problemem jest dynamicznie zmieniający się promień okręgu i właśnie do znalezienia najlepszego promienia użyto algorytmów genetycznych. W metodzie prezentowanej w artykule każdy osobnik złożony jest z wektora cech. Cechy są liczbami rzeczywistymi o odpowiadają współrzędnym wyszukiwania i promieniowi znaku. Dodatkowo algorytm zawsze replikuje najlepsze rozwiązanie. W artykule również proponują modyfikacje algorytmu genetycznego nazwaną Step-GA starającą się testować kolejne pokolenia na obrazach wybranych sekwencji filmowej z ustaloną częstotliwością. By poprawić zdolności wykrywania znaków przez algorytm autorzy zaproponowali prostą metodę śledzenia znaku. W większości przypadków podczas jazdy samochodem znak drogowy wykryty na środku ekranu będzie się powiększał i przesuwał w okolice prawego górnego rogu. Jeżeli algorytm wykryje obszar, który będzie zachowywał się

w taki sposób dopiero wtedy może być on kwalifikowany jako znak drogowy. Takie działanie pozwala odfiltrować dużą część regionów w których wystąpiły błędy drugiego rodzaju. Jednak mimo dużej ilości zalet metody w trakcie jej publikacji czyli roku 2002 była dość niedoskonała. Autorzy nie testowali swojego algorytmu na żadnym międzynarodowym benchmarku dlatego ciężko ocenić skuteczność przedstawionej metody.

W artykule [13] również zaproponowano metodę wykrywania znaków bazującą na algorytmie genetycznym. Analogicznie jak w poprzedniej pracy algorytm stara się odnaleźć pozycję i wielkość znaku. Różnicą jest uwzględnienie również rotacji znaku. Zazwyczaj w GA populacja początkowa generowana jest losowo. W pracy zauważono, że można poprawić wynik algorytmów generując populację początkową z określonego regionu. Generowanie populacji poprzedzone jest progowaniem obrazu w czerwonej barwie co pozwala uzyskać informację w jakich regionach znak ma większe prawdopodobieństwo wystąpienia. Rysunek 6 przedstawia cztery etapy inicjalizacji algorytmu genetycznego wspomaganą wiedzą o występowaniu obszarów czerwonych.

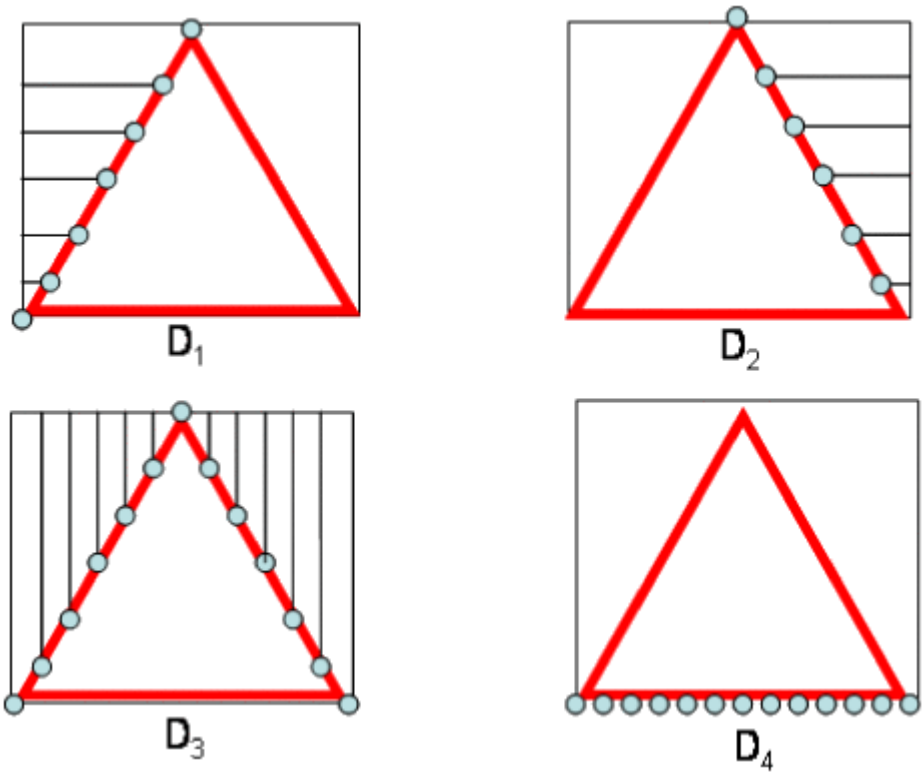


Rysunek 6 Inicjalizacja algorytmu GA wspomagana progowaniem

Na obrazie (a) pokazany został oryginalny obraz wejściowy. Obraz (b) przedstawia progowanie w kolorze czerwonym. Obraz (c) przedstawia filtrację obrazu co pozwala uzyskać tylko regiony najbardziej pożądane. Ostatni obraz (d) przedstawia regiony gdzie znak drogowy ma największe prawdopodobieństwo wystąpienia. Czas pracy generowania każdego pokolenia wynosi 8.8 ms pracując na procesorze AMD Duron 1 GHz. Po znalezieniu regionów występowania znaku rozpoznanie znaku odbywa się za pomocą sieci neuronowej nauczonej przez obrazy dostarczone w przestrzeni HSI. Warto wspomnieć, że prace dotyczące algorytmów genetycznych w połączeniu z sieciami neuronowymi stosowane były od początku rozważań nad tematem TSR. Przykładem może być przytoczenie pracy [1] z 1996 roku która proponowała takie rozwiązanie.

SVM

Maszyna wektorów nośnych SVM (ang. Support Vector Machine) jest klasyfikatorem umożliwiającym określenie do jakiego zbioru należy obiekt. Maszyna do poprawnego działania potrzebuje zbioru uczącego, w którym przedstawione są poprawnie sklasyfikowane obiekty. SVM znajduje hiperpłaszczyznę oddzielającą dwie klasy. Jeżeli taka funkcja nie istnieje w przestrzeni zadania należy wprowadzić funkcję jądra, która powiększy przestrzeń. System, który korzysta z maszyny wektorów nośnych dla problemu TSR został przedstawiony w pracy [11]. Działanie systemu polega na wykrywaniu obszary występowania znaków za pomocą progowania w przestrzeni HSI. Znaki białe wykrywane są za pomocą rozkładu achromatycznego. Każdy obszar prawdopodobnego wystąpienia znaku wpisywany jest w kwadrat. Następnie wyznacza się 20 odległości liczonych od ramki kwadratu do znalezionego koloru w środku ramki mający być obwiednią obiektu pokazanych na Rysunek 7 Odległości tworzące wektor DtB. Z tych 20 odległości tworzy się wektor DtB dostarczany jest na wejście SVM.



Rysunek 7 Odległości tworzące wektor DtBs

Warto wspomnieć, że wyznaczony w ten sposób wektor znaków ośmiokątnych niezbyt różni się od znaków okrągłych dlatego znaki ośmiokątne klasyfikowane są dopiero na etapie rozpoznawania znaku. Następnie przy użyciu kilku wytrenowanych maszyn SVM następuje klasyfikacja obszaru. Każda maszyna potrafi rozpoznać jeden typ obrazu. W zależności od wykrytego koloru (czerwonego, niebieskiego, żółtego lub białego) algorytm wyznacza inne maszyny do klasyfikacji. Proces rozpoznania opiera się na maszynie SVN z ziarnami Gaussa. Dużą zaletą metody jest jej odporność na rotację.

HOG i SVM

Maszyna wektorów nośnych może być również łączona z algorytmem tworzącym histogram gradientów zorientowanych. Takie podejście zaprezentowane zostało w pracy [16]. W tej pracy

wykrywanie obiektów za pomocą metody HOG wspomagane jest informacją o kolorze, co jest ogromnie istotne dla wykrywania znaków drogowych. Oryginalnie algorytm HOG oblicza gradienty dla każdego kanału kolorów i przyjmuje gradient o największej normie, zaś algorytm w opisywanej pracy [16] oblicza funkcje HOG dla każdego kanału kolorów po czym łączy je ze sobą tworząc histogram. W odróżnieniu od podobnych metod, obliczenie funkcji HOG następuje na podstawie mapy prawdopodobieństwa, ponieważ mapa prawdopodobieństwa może kodować informacje o kolorze i kształcie znaku drogowego, jednocześnie tłumiąc wpływ tła. Jednak na podstawie samej mapy prawdopodobieństwa nie ma możliwości zidentyfikowania znaku. Aby rozwiązać ten problem, dodano dodatkową funkcję w postaci klasycznego algorytmu HOG. Do identyfikacji przeszkolony został wielopoziomowy klasyfikator SVM. Autorzy algorytmu przetestowali swoją pracę na zbiorze danych GTSDB, gdzie istnieją trzy kategorie znaków drogowych, dlatego wyszkolony został 4-klasowy klasyfikator SVM z dodatkową klasą tła. Za jądro klasyfikatora SVM została wybrana funkcja radialna, ponieważ najlepiej sprawdziła się w eksperymentach. Zaproponowany algorytm na bazie danych GTSDB wykrywa znaki zakazu i nakazu z 100 % poprawnością, znaki ostrzegawcze z 94.29 % poprawnością. Czas wykrycia znaków w ramce to 0,067 sekundy co jest bardzo zadowalającym wynikiem.

Inne podejścia

Algorytm Violi i Jonesa

Algorytm Viola–Jones zaproponowany w 2001 przez Paul Viola i Michael Jones wprawdzie został stworzony do wykrywania twarzy [24], lecz znakomicie nadaje się do wykrywania obiektów każdego typu na obrazie. Podejście dla problemu TSR zostało przedstawione w pracy [2]. W pracy jako klasyfikatora używa się algorytmu opartego na boostingu nazwanego AdaBoost (ang. Adaptive Boosting). Tworzy on z kilku mniejszych klasyfikatorów jeden silny klasyfikator. Dla ww. pracy jako słabe klasyfikatory przyjmuje się drzewa decyzyjne, w których każdy węzeł jest jedną z cech haaro-podobnych. Cechy Haara swoją nazwę zawdzięczają falką Haara zaproponowanym przez Alfréda Haara w 1909 lub 1910 roku. Cechy Haara to najprościej mówiąc pewna maska, która posiada dwa typy pikseli. Po jej nałożeniu na część obrazu sumuje się jasności pikseli należących do danego typu. Jeżeli różnica między dwoma typami jest wystarczająca cecha Haara daje pozytywną odpowiedź. Algorytm używa niewielkiej ramki przesuwanej po obrazie ROI. W każdym położeniu ramki za pomocą cech Haara stara się odnaleźć miejsce występowania znaku. Cechy Haara sprawdzane są kaskadowo co pozwala od razu odrzucić rejony gdzie znak nie występuje. By przyspieszyć obraz zostaje scałkowany. W wcześniej już wspomnianej pracy [2] metoda pozwoliła na wykrywanie nie tylko znaków drogowych lecz również na wykrycie samochodów a nawet rowerzystów. Ponieważ algorytm oprócz wykrywania znaków wykrywa dwa inne typy obiektów, wprowadzono dodatkowo funkcje subkategoryzacji. Wielką zaletą metody jest wykrywanie obiektów w czasie rzeczywistym. Praca, która korzysta z metody opartej na AdaBoost i cechach Haara została zaprezentowana w [25]. Dzieło pozwala na wykrywanie i śledzenie obiektów. Cechy Haara zostały również wsparte informacją o kolorach co pozwoliło na redukcję błędów pierwszej klasy z 1.6 % do 1.4 %, oraz błędów drugiej klasy z 0.3 % do 0.03 %, co jest spektakularnym wynikiem. Praca [10] pozwala na śledzenie obiektów i również wykorzystuje kaskadę AdaBoost. Co ciekawe algorytm przedstawiony w pracy jest w stanie sam się doszkalać przez zbieranie pozytywnych i negatywnych próbek. Algorytm do doszkalania wykorzystuje śledzenie wsteczne. Oczywistością jest, że znaki które są bliżej są

również lepiej rozróżnialne, dlatego po wykryciu znaku algorytm sprawdza poprzednie klatki filmu w poszukiwaniu wykrytego znaku w mniejszej skali i stara się samoczynnie doszkolić.

Symetria promieniowa
Niektóre ostatnie prace [26] i [27] wdrożyły szybki algorytm oparty na symetrii promieniowej, która dostosowany do kształtów trójkątnych, kwadratowych, diamentowych, ośmiokątnych i okrągłych. Algorytm uwzględnia gradient obrazu dostarczonego skali szarości i wykorzystuje naturę kształtów, które głosują w punkcie środkowym dla okrągłych znaków i linii głosów w przypadku regularnych wielokątów. Główną zaletą tej metody jest to, że jest ona w stanie działać w czasie rzeczywistym. Ponieważ wykrywa kształty oparte na krawędziach, algorytm jest odporny na zmiany oświetlenia.

Klasyfikacja znaków drogowych

Klasyfikowanie znaków, wyłączając etap śledzenia, jest ostatnim etapem wykrywania znaków drogowych. Omawiana poprzednio segmentacja zawęża się obszar występowania znaku co w rezultacie prowadzi do faktu, że znak zajmuje około 70 % fragmentu odnalezionego przez segmentacje. Zadaniem klasyfikacji jest odróżnienie czy w miejscu wskazanym przez segmentacje znak rzeczywiście istnieje i jeżeli tak jest odczytanie znaku. Obecnie najpowszechniej stosowanym klasyfikatorem obrazów są sieci neuronowe. Warto jednak wspomnieć, że prócz sieci istnieją inne metody rozpoznawania typu znaku. W tej części zostały opisane metody klasyfikacji na potrzeby systemu TSR.

Template matching

Template matching jest klasyczną metodą bazującą na informacji wzajemnej pozwalającą odnaleźć obiekt na obrazie. Na potrzeby metody TM wzorce obiektów, które należy rozpoznać gromadzone są w bazie wzorców. Następnie zazwyczaj prostą metodą polegającą na dopasowaniu dwóch obrazów do siebie np. metodą najmniejszych kwadratów. Metoda TM jest prosta w zaimplementowaniu lecz przez czas pracy dla kilkudziesięciu typów znaków drogowych jest nieefektywna. Szczegółowy opis metody można znaleźć w publikacji [28]

Sztuczne sieci neuronowe

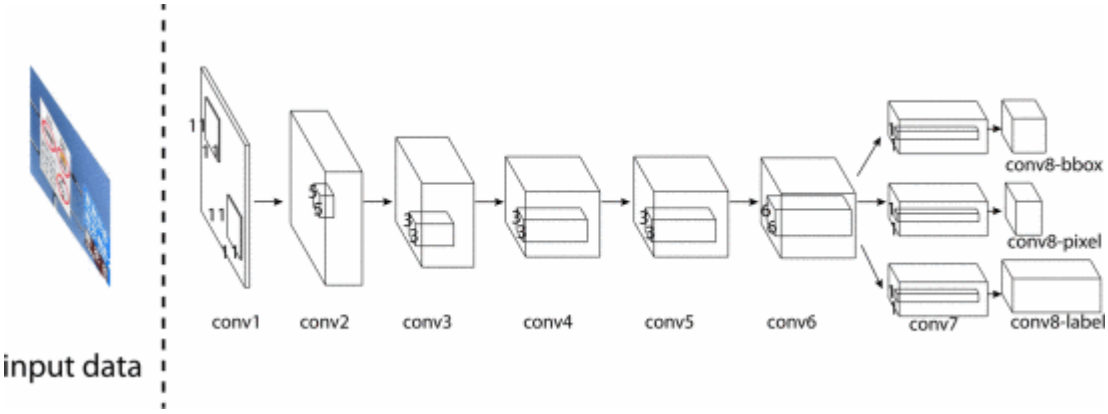
Sztuczne sieci neuronowe są zaprogramowanymi strukturami, które symulują model zdolny do wykonywania obliczeń matematycznych. Idea sztucznych sieci neuronowych inspirowana była budową ludzkich neuronów. Każda sieć zbudowana jest z warstw węzłów zwanych neuronami. Węzły połączone są ze sobą synapsami. Zazwyczaj do jednego węzła wchodzi wiele synaps. Każda synapsa posiada swoją wagę, a węzeł funkcję aktywacji. Gdy na wejściu synapsy pojawia się sygnał mnożony on jest przez wagę synapsy i podawany na węzeł. Gdy łączna wartość sygnału podanego na węzeł jest wystarczająca sygnał podawany jest dalej na kolejną warstwę. Sieci neuronowe stosowane są do wszelakich zadań lecz na potrzeby pracy zostanie przedstawione zastosowanie w rozpoznawaniu obrazów. W takim zastosowaniu na wejściu sieci podawane są piksele obrazu, który należy rozpoznać. Zazwyczaj sieć ma określoną liczbę pikseli wejściowych, dlatego wykryty obszar przez segmentację należy przeskalować i znormalizować do tej liczby. Do normalizacji obrazu można stosować wiele metod. Najprostszą z nich jest odrzucenie co pikseli, co konkretną wartość liczbową . Rozwiązaniem dzięki któremu traci się mniejszą ilość informacji lecz jest bardziej kosztowny obliczeniowo jest interpolacja dwuliniowa. Już w artykule z 1996 roku [17] zostały użyte sieci neuronowe do klasyfikacji typu znaku drogowego. Istnieje ogromna liczba typów sieci neuronowych i metod usprawniających ich działanie. Sieć składająca się z więcej niż jedna warstwa ukryta

nazywana jest siecią wielowarstwową. Przykładem takiej sieci jest perceptron wielowarstwowy.

MLP

MLP (ang. Multilayer perceptron) - perceptron wielowarstwowy jest najpopularniejszy rodzajem sieci neuronowych. Taka sieć składa się z jednej warstwy wejściowej i kilku warstw ukrytych. Na potrzeby klasyfikacji w węzłach sieci MLP znajduje się funkcja nielinowa. Typem sieci MLP służącym do rozpoznawania obrazów jest splotowa sieć neuronowa inaczej znana jako sieć konwolucyjna. Tego typu sieć została przedstawiona w wcześniej wspomnianej pracy [17]. Autorzy użyli dwóch sieci typu MLP. Jednej dla znaków trójkątnych, a drugiej dla okrągłych. Wielkość wejścia sieci odpowiadała obrazowi o rozmiarach 30x30 piksel. Warstwa wyjściowa sieci posiadała 10 wyjść z czego 9 wyjść odpowiadało znakom. Ostatnie wyjście informowało, że znak nie został rozpoznany. Autorzy eksperymentalnie dobrali wielkość sieci z pośród 3 typów sieci. Najlepsze wyniki uzyskała 3 konfiguracja 30x30/15/5/10 z 98% skutecznością dla najlepiej rozpoznawalnych znaków. Rozpoznawanie na jednostce obliczeniowej PC486 33 MHz zajmuje 220 ms dla obrazu w rozdzielczości 256x256 co jest znakomitym wynikiem jak na rok 1996. Należy jednak zaznaczyć, że segmentacja w tej pracy była bardzo kosztowna czasowo, co nie pozwoliło algorytmowi działać w czasie rzeczywistym.

W innej pracy przedstawionej w 2003 roku [13] również użyto splotowych sieci neuronowych do rozpoznawania znaków drogowych. W tej pracy segmentacja obszaru występowania znaku przeprowadzona była za pomocą progowania oraz algorytmów genetycznych. Sieć była uczona za pomocą algorytmu wstecznej propagacji błędu (ang. Backpropagation Neural Network), według paradygmatu ART1 (ang. Adaptive Resonance Theory). Sieć posiada dwie warstwy ukryte. Dla całego algorytmu przedstawionego w pracy rozpoznanie obrazu wraz z klasyfikacją znaków zajmuje mniej niż 8.8 ms na procesorze AMD Duron 1 GHz. Biorąc jednak pod uwagę, że dla systemów TSR rozpoznanie w czasie rzeczywistym liczone jest od prędkości 0.2 s dla jednego obrazu wynik pracy jest bardzo dobry. W pracy [29] opracowano algorytm bazujący na sieci neuronowe propagacji wstecznej pozwalający na śledzenie obiektów. Na potrzeby algorytmu stworzono dwie sieci działające równolegle ze sobą. Jedna z nich stara się zidentyfikować obiekt na bazie koloru, a druga na bazie wykrytych krawędzi. Obiekt jest rozpoznawany i klasyfikowany a następnie śledzony. Czas rozpoznania obrazu wynosi 0.3 s. Sieć neuronowa uczona algorytmem wstecznej propagacji została również zaimplementowana w 2005 r. przez autorów artykułu [30]. Autorzy wspomnieli, że proces uczenia sieci algorytmem wstecznej propagacji jest kosztowny pod względem obliczeniowym oraz łatwo podczas szkolenia sieci wpaść w lokalne minimum co przysparza problemów. By zapobiec ostatniemu problemowi część autorów algorytmów starała się uczyć sieć metodami głębokiego uczenia DL (ang. Deep Learning). Warto zauważyć, że przy uczeniu sieci jakąkolwiek metodą ważne jest by niektóre obrazy uczące były w pewien sposób zaszumione oraz obrócone pod małym kontem. Taki zabieg powoduje większą odporność sieci. W pracy [31] szkolono dwie sieci wielowarstwowe metodą głębokiego uczenia. Jedna sieć była przeznaczona do segmentacji obrazów, a druga do klasyfikacji. Obie sieci posiadały podobną strukturę z wyjątkiem ostatniej warstwy. Przy testowaniu sieci przeznaczonej do segmentacji obrazu zauważono, że sieci splotowe są znacznie wydajniejsze gdy zostają użyte w trybie przesuwanego okna. Poza tym część informacji z nakładających się regionów można użyć wielokrotnie co przyspiesza algorytm. Architektura sieci przedstawiona jest na Rysunek 8 Architektura sieci splotowej z rozgałęzieniem po 6 warstwie

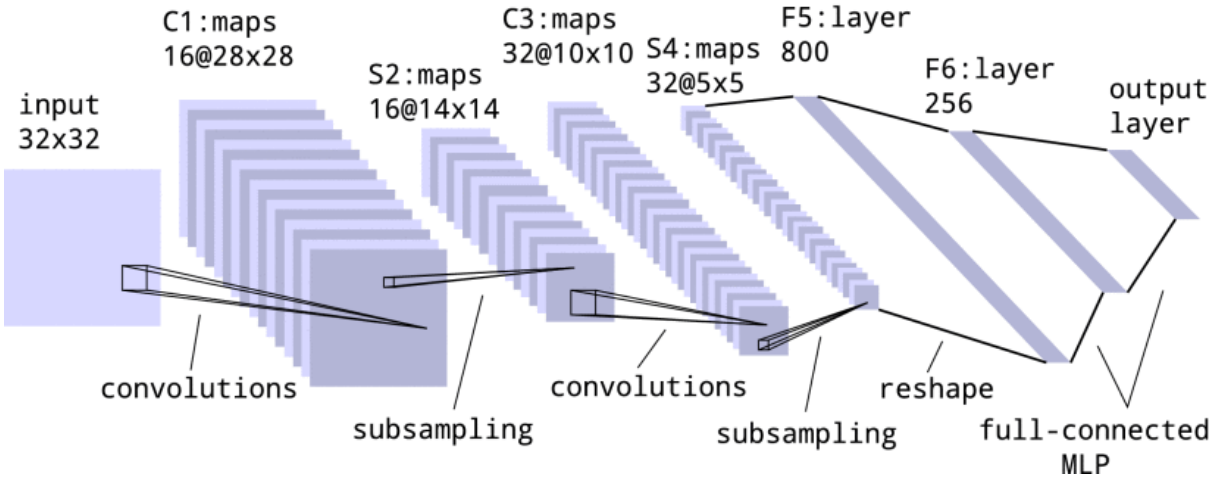


Rysunek 8 Architektura sieci splotowej z rozgałęzieniem po 6 warstwie

Przy szkoleniu sieci niektóre próbki szkoleniowe były zniekształcone o ± 20˚, oraz dodano obrazy nie zawierające znaków drogowych. Łączenie użyto około 6600 obrazów do szkolenia i 3400 do testowania sieci. Do testów została użyta jednostka Intel Xeon E5-1620, dwoma kartami NVIDIA Tesla K20 G-PU i 32 GB pamięci. Sieć wykrywa znaki drogowe w ciągu 0.8 s z 88 % skutecznością.

CNN

CNN (ang. Convolutional Neural Network) czyli splotowe sieci neuronowe, nazywa się takie sieci, których pierwsze warstwy dokonują splotu danych wejściowych. Takie sieci zostały wprowadzone by zapobiec problemowi przeuczenia się sieci. Splot może polegać na łączeniu informacji z kilku pikseli i przekazywaniu jej jako ujednolicona wartość. W pracy [16] zastosowano splotową sieć neuronową dla by sklasyfikować znaki drogowe. W tym celu wyszkolono trzy siec, każda dla innego typu znaku. Na wejście dostarczano obrazy w skali szarości i rozmiarze 32×32 pikseli. Architektura sieci przedstawiona jest na Rysunek 9



Rysunek 9 Struktura sieci splotowej opisywanej pracy

Ponieważ obrazy są przechwytywane w różnych warunkach oświetleniowych i pogodowych, znaki tej samej klasy mogą znacząco różnić się od siebie. Aby zmniejszyć różnicę jasności użyto metody korelacji adaptacyjnego histogramu ograniczonego kontrastem (CLAHE), pozwalającej dostosować kontrast obrazów. Z powodu tego, że wykryte regiony mogą zawierać błędy pierwszego i drugiego rodzaju dodano klasę tła w trakcie uczenia. Każda z sieci zawierała dwie warstwy splotowe i dwie warstwy posiadające architekturę sieci MLP. Rozmiar

obrazu wejściowego wynosi 32×32. Rozmiar filtru w obu warstwach splotowych wynosił 5×5. Po pierwszej warstwie splotu, zostało dodanych się 16 map powodując zmniejszenie rozmiaru wejścia drugiej warstwy splotowej najpierw do rozdzielczości 28×28, a następnie do 14×14. Algorytm po transformacjach obrazu i zmapowaniu dostarcza wektor o długości 800 znaków. Sieć jest zaprogramowana w języku C++ z wykorzystaniem biblioteki OPENMP. Wykorzystywane są również implementacje MSER, HOG i SVM z biblioteki OpenCV. Sieć neuronową wyszkolono za pomocą Torch7 i przepisano wyszkolone wartości do kodu C++. Algorytm został uruchomiony na komputerze z czterordzeniowym procesorem 3,7 GHz, poprawnie wykrywając z dokładnością 99.29 % znaków zakazu, 96.74 % znaków nakazu i 97.13 % znaków ostrzegawczych przy czasie wykrywania wynoszącym 0.162 s. Znaki do testowania wzięto z niemieckiego benchmarku GRSDB.

Architektura sieci splotowej przedstawionej w artykule [14] różni się od tradycyjnych sieci splotowych typem nieliniowości, ponieważ w węzłach sieci użyto funkcji przypominającą funkcję tangensa hiperbolicznego. Dodatkowo wykorzystano połączenia między niesąsiadującymi warstwami sieci. Tradycyjne sieci splotowe nie posiadają połączeń między warstwami niesąsiadującymi. W warstwie poolingu użyto różnych parametrów odpytywania dla danych dostarczanych z różnych warstw. Sieć została zaimplementowana z użyciem biblioteki EBLearn, oraz była również testowana na znakach pochodzących ze zbioru GTSRB. W przeciwieństwie do poprzedniej sieci ta sieć była uczona dla wszystkich zbiorów znaków jednocześnie. Co ciekawe sieć osiągnęła wynik 98.97 % dokładności wykrywania dla architektury, w której zostały wykorzystane informację o kolorach i 99.17 % dokładności dla obrazów w skali szarości. Jest to sprzeczne z ogólnym poglądem twierdzącym, że wykorzystanie kolorów powinno poprawić skuteczność algorytmu.

ELM

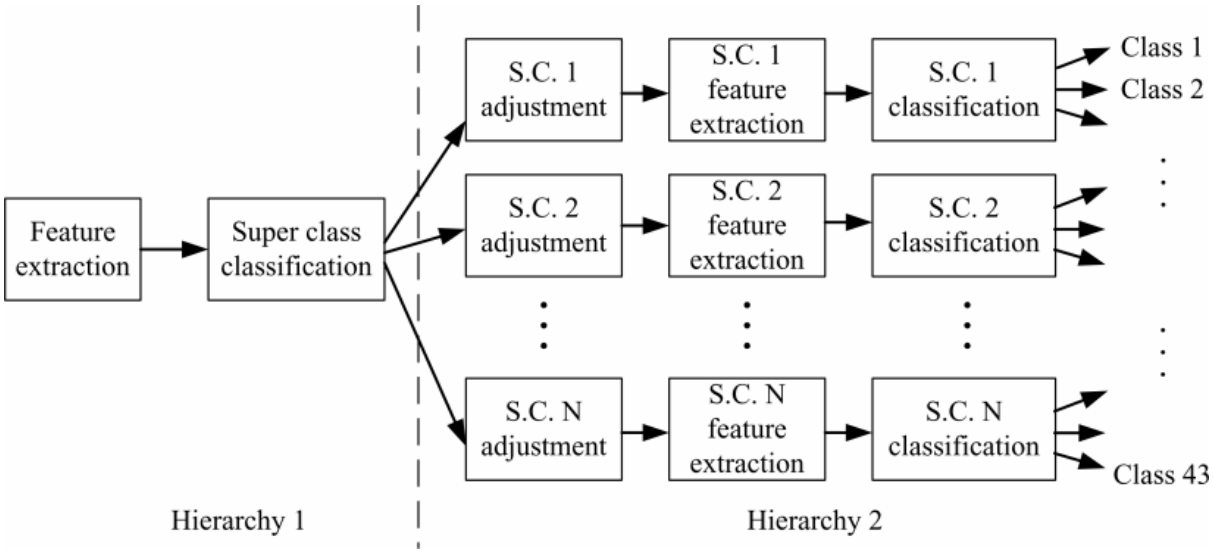
ELM (ang. Extreme Learning Machine) jest algorytmem uczenia się pojedynczych sieci ukrytych sieci neuronowej (SFNN). Pierwszą zaletą algorytmu ELM jest to, że wartości wejściowe między wejściem i warstwami ukrytymi są losowo przypisywane, co pozwala losowe mapowanie cech. Ponieważ wyszkolone są tylko warstwy wyjściowe między warstwami ukrytymi i wyjściowymi, nie jest wymagane strojenie za pomocą wstecznej propagacji warstwa po warstwie. Algorytm ELM może uzyskać optymalne i uogólnione rozwiązanie do rozpoznawania wieloklasowego. Dodatkowo, łatwo można rozszerzyć ELM na sieć wielowarstwową lub głęboką za pomocą techniki autoencoder. Metoda ELM została również wykorzystana do modelowania lokalnych pól recepcyjnych i wykorzystywana do uczenia na dużych zbiorach danych . W związku z tym zastosowanie ELM dla TSR może dać lepsze rozwiązanie. Co więcej, z powodu losowego przypisania wag wejściowych, algorytm ELM może zmniejszyć koszt obliczeniowy szkolenia. Ponieważ istnieje tylko jedna ukryta warstwa, szybkość obliczeniowa procesu rozpoznawania jest również szybka. Praca [22] stosuje jako klasyfikator pojedynczą sieć z ukrytą warstwą. W oparciu o algorytm ELM, połączenie pomiędzy warstwami wejściową i ukrytą realizuje mapowanie cech losowych, podczas gdy tylko wagi pomiędzy warstwami ukrytymi i wyjściowymi są wyszkolone. W rezultacie strojenie warstwa po warstwie nie jest wymagane. Tymczasem norma wag wyjściowych jest zawarta w funkcji kosztów. W związku z tym, Klasyfikator oparty na ELM może osiągnąć optymalne i uogólnione rozwiązanie dla wieloklatkowych TSR. Ponadto może zrównoważyć dokładność rozpoznawania i koszty obliczeniowe.

LDA

LDA (ang. Linear Discriminant Analysis) - liniowa analiza dyskryminacyjna jest to algorytm, którego używa się w uczeniu maszynowym do znalezienia liniowej kombinacji cech, które najlepiej rozróżniają dwie lub więcej klas obiektów lub zdarzeń. Wynikowe kombinacje są używane jako klasyfikator liniowy lub, częściej, służą redukcji wymiarów do późniejszej klasyfikacji statystycznej. [here] (usunąć cały opis??)

SVM

W artykule [19] zaproponowano innowacyjną metodę rozpoznawania znaków nazwaną hierarchiczną metodą wektorów nośnych. Algorytm zajmuje się wyłącznie klasyfikacją znaków drogowych bez etapu segmentacji i był uczony na zbiorze GTSRB. Algorytm działa w następujący sposób. Wpierw znak drogowy klasyfikowany jest przez SVM do jednej z trzech superklas. Następnie w zależności do jakiej klasy znak drogowy został zakwalifikowany zostaje użyta funkcja dostosowania perspektywy z różnymi parametrami. Po dostosowaniu perspektywy ponownie zostają użyte maszyny wektorów nośnych zdolne rozpoznać dokładnie co jest przedstawione na znaku. W roli wyjaśnienia jedna maszyna klasyfikuje znaki do podklas, a dla każdej podklasy wytrenowane są inne maszyny zdolne do odczytania znaku. Pierwsza maszyna SVM przydziela klasę znaku z pomocą algorytmu HOG. Dzięki rozpoznaniu superklasy znaku można wykorzystać informację o kolorze. Skorygowanie perspektywy jest kluczowym elementem pomagającym lepiej rozpoznać znak drogowy. Taka czynność eliminuje wszystkie translacje i rotację znaku co pozwala lepiej odczytać znak w następnym etapie. Schemat blokowy proponowanej metody znajduje się Rysunek 10.



Rysunek 10 Schemat blokowy metody opartej na SVM

Algorytm dokonuje wstępnej ekstrakcji cech obrazu za pomocą wyszukania czerwonych pikseli oraz sprawdzeniu pikseli sąsiadujących z czerwonymi pikselami. Po wstępnym przetworzeniu obrazu następuje znalezienie okręgów na obrazie za pomocą radialnego detektora symetrii. Radialny detektor symetrii jest wariantem kołowej transformaty Hougha. Po wykryciu okręgów następuje wykrycie trójkątów czyli znaków ostrzegawczych. Do wykrywania trójkątów na obrazie również użyta zostaje transformata Hougha wykrywająca linie. Następnie algorytm wykrywa linie przecinające się mogące stanowić kontury znaku. Na chwilę obecną algorytm zaproponowany w pracy z 43 klas znaków dostępnych w GTSRB rozróżnia znaki ostrzegawcze i zakazu będące okręgami i trójkątami z dokładnością 99.03 %.

Znaki należące do innych klas są klasyfikowane za pomocą algorytmu SVM i HOG bez informacji o kolorze z dokładnością 99,94%, 98,89% i 99,90% dla znaków kolejno nakazu, ograniczeń prędkości i znaków pozostałych, co daje globalny wynik dokładności algorytmu na poziomie 99,52% z średnią szybkością 40 ms. Implementacja została wykonana za pomocą programy Matlab 2011b i na jednostce Core 13 3.3 GHz.

Praca [11] proponuje kompletny algorytm bazujący na SVM pozwalający zlokalizować i rozpoznać znak drogowy. Wcześniej zaznaczono, że segmentacja dokonywana jest w przestrzeni HSI, a regiony wykryte przez segmentację wstępnie są przydzielane do podklas za pomocą linowych maszyn SVM. Pozwala to wstępnie rozróżnić typ znaku. Rozpoznawanie realizowane jest przez maszyny SVM z jądrami Gaussa. Praca do treningu wykorzystuje bibliotekę LIBSVMS. Starano się użyć funkcji liniowej jako hiperpłaszczyzny lecz nie zawsze było to możliwe. W trudnych przypadkach wprowadzono jądro Gaussa. Na wejście SVM podawano 31x31 pikseli dlatego każdy obszar gdzie zostały wykryte potencjalne znaki musiał być przeskalowany do tej rozdzielczości. Dla każdego koloru i kształtu podano od 20 do 100 próbek uczących. Przetwarzanie pojedynczej ramki o wielkości 720×576 pikseli za pomocą proponowanej metody zajmuje 1.77 s na procesorze Pentium 4-M 2,2 GHz. Testy prowadzone były na autorskim benchmarku znaków obowiązujących w Hiszpanii.

Drzewo K-d

Praca [32] jest przykładem artykułu starającym się rozważyć przydatność drzew wielowymiarowych K-d (ang. K-Dimensional Tree) do rozróżniania znaków drogowych. Drzewo K-d jest wariantem drzewa binarnego opartego o wyszukiwanie najbliższego sąsiada. Zaletą drzew K-d jest łatwość budowy i aktualizacji oraz szybkość ich przeszukiwania. Dodatkowo drzewa dobrze radzą sobie z niezrównoważonymi zbiorami danych w przeciwieństwie do sieci neuronowych, które wymagają dobrego dostrojenia parametrów. W przypadku pracy drzewo K-d składa się z informacji dostarczonych od czterech transformat HOG na obrazie, każdej z innymi parametrami. Do wyszukiwania najbliższego sąsiada użyto algorytmu Best Bin First. Na znakach dostarczonych przez zbiór danych GTSRB drzewo K-d bazujące na deskryptorze HOG przedstawione w pracy uzyskało poprawność rozpoznania na poziomie 92,9%.

RM

W tej samej pracy co drzewa K-d [32] testowano również lasy losowe RM (ang. Random Forests) do problemu klasyfikacji znaków drogowych. Klasyfikacja obrazu przez las losowy polega na zebraniu odpowiedzi z kilku drzew decyzyjnych przez głosowanie. Lasy losowe również jak drzewa D-d dobrze radzą sobie z klasyfikacją na zbiorze niesymetrycznym niekiedy przewyższając niekiedy algorytmy SVM. Jak i w drzewie K-d opisywanym powyżej obraz wejściowy poddawany był 4 transformatą HOG. Dla lasu losowego testowanego na tym samym zbiorze danych GTSRB poprawność klasyfikacji została poprawiona do 97,2%.

Podsumowanie

Praca ma na celu zbadanie w jaki sposób zastosowanie wielu kamer może polepszyć wykrywanie znaków drogowych. W podrozdziale odnoszącym się do łączenia obrazów w panoramę zostało omówionych kilka metod, z czego najskuteczniejsze okazały się metody bazujące na punktach charakterystycznych. Są one szybkie w działaniu oraz odporne na małe rotacje dwóch obrazów. Należy również zaznaczyć, że problem tworzenia panoramy na potrzeby pracy niesie ze sobą wiele udogodnień. Z dość dużą dokładnością znana jest translacja

pozioma obu obrazów. Translacja pionowa praktycznie nie występuje, co prowadzi do wniosku znany jest obszar w którym obrazy z kamer się nakładają. Umożliwia to wykonywanie operacji dopasowania tylko na wspólnym obszarze rejestrowanym przez obie kamery co pozwoli zmniejszyć ilość obliczeń. Przez wprowadzone ograniczenia metody klasyczne nie są od skazane na niepowodzenie. Metody bazujące na punktach charakterystycznych wydają się przynosić najbardziej obiecujące wyniki.

Podsumowując zagadnienie segmentacji należy stwierdzić, że na dzień dzisiejszy podejścia oparte na samym kolorze wyszły z użycia, a najpowszechniejsze są algorytmy hybrydowe. Duża część algorytmów korzysta z połączenia podejść opartych na kolorze i kształcie bądź kolorze i teksturze. Są one szybkie i skuteczne. Podejścia oparte na teksturze (HOG, LPB, ACF) z dużą dokładnością potrafią wyłonić obrys znaku drogowego, co na etapie identyfikowania może posłużyć do szkolenia klasyfikatora. Niestety takie przetwarzanie na całym obrazie jest zbyt kosztowne obliczeniowo, dlatego zawęża się obszar stosując ROI. Pozwala to przyspieszyć obliczenia i umożliwia przetwarzanie obrazu w czasie rzeczywistym. Obszar ROI dotyczy głównie miejsc środkowych i po prawej stronie. By lepiej wyznaczyć położenie ROI można zastosować progowanie w określonym kolorze, najczęściej w czerwonym. Podczas progowania najczęściej zostają odnalezione grupy pikseli nie powiązane ze sobą. By zlokalizować obiekty odnalezione przez progowanie należy w jakiś sposób połączyć piksele ze sobą i je oznaczyć. Dość prostą i skuteczną metodę przedstawiono w pracy [9] gdzie do każdego odnalezionego piksela zostaje przypisana etykieta. Algorytm etykietowania wprowadza osobny bufor umożliwiający łączenie regionów z różnymi etykietami. Niektóre algorytmy jak ten zaproponowany w pracy [33] używają funkcji ACF z klasyfikatorem AdaBoost. Algorytm ACF został również użyty w pracy [34] w połączeniu z informacją o kolorach i gradientach oraz algorytmem HOG i uzyskał dobre wyniki. Oprócz powyższych podejść, splotowa sieć neuronowa (CNN) jest stosowana do wykrywania znaków drogowych i osiąga doskonałe wyniki [14]. Warto również wspomnieć, że dla lepszej wstępnej filtracji obrazu można zastosować również Filtry Gabora lecz obecnie używanie takiego algorytmu jest nieefektywne ze względu na korzyści i czas obliczeń. Wartym wspomnienia podejściem detekcji znaków jest tworzenie mapy cieplnej obszarów występowania znaków [35]. Taka mapa działa na zasadzie algorytmów biologicznych. Warto również dodać, że część obliczeń wykorzystanych w etapie tworzenia panoramy może posłużyć w etapie segmentacji obiektów.

Najpowszechniej stosowanym podejściem do klasyfikacji obiektów są splotowe sieci neuronowe. Czas klasyfikacji odnajdywania obiektów przez sztuczne sieci neuronowe balansuje na granicy pracy w czasie rzeczywistym dlatego niektórzy autorzy starają się przyspieszyć wykrywanie znaków drogowych łącząc sieci CNN z innymi algorytmami np. w pracy [16] algorytmem SVM. Drzewa wielowymiarowe oraz lasy losowe chodź osiągają znakomite wyniki niestety okazuje się, że są zbyt wole by pracować w czasie rzeczywistym.

Program

Propozycja rozwinięcia/konstrukcji metody

Przygotowanie środowiska badawczego i plan badań

Podsumowanie

Bibliografia

[1] Y. Aoyagi i T. Asakura, „A study on traffic sign recognition in scene image using genetic algorithms and neural networks," w *International Conference on Industrial Electronics, Control, and Instrumentation*, Taipei, Taiwan, Taiwan, 9 sierpień 1996.

[2] Q. Hu, S. Paisitkriangkrai, C. Shen, A. v. d. Hengel i F. Porikli, „Fast Detection of Multiple Objects in Traffic Scenes With a Common Detection Framework," *IEEE Transactions on Intelligent Transportation Systems,* tom 17, nr 7, pp. 1002-1024, kwieceń 2016.

[3] A. Mogelmose, M. M. Trivedi i T. B. Moeslund, „Vision-Based Traffic Sign Detection and Analysis for Intelligent Driver Assistance Systems: Perspectives and Survey," *Transactions on Intelligent Transportation Systems,* tom 13, nr 4, pp. 1484-1497, grudzień 2012.

[4] B. Zitová i J. Flusser, „Image registration methods, a survey. Image and Vision Computing," *Image and Vision Computing,* tom 21, nr 11, p. 977–1000, październik 2003.

[5] F. Maes, A. Collignon, D. Vandermeulen, G. Marchal i P. Suetens, „Multimodality image registration by maximization of mutual information," *Transactions on Medical Imaging,* tom 16, nr 2, p. 187–198, kwiecień 1997.

[6] M. Kondej, B. Putz i M. Bartyś, „Szybki algorytm dopasowania obrazów dla potrzeb fuzji w czasie rzeczywistym," 25 listopad 2010. [Online]. Available: https://automatykaonline.pl/Artykuly/Sterowanie/szybki-algorytm-dopasowania-obrazow-dla-potrzeb-fuzji-w-czasie-rzeczywistym. [Data uzyskania dostępu: 11 03 2019].

[7] J. Heather i M. Smith, „New adaptive algorithms for real-time registration and fusion of multimodal imagery," 5 maj 2010. [Online]. Available: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/7689/768908/New-adaptive-algorithms-for-real-time-registration-and-fusion-of/10.1117/12.850029.short?SSO=1. [Data uzyskania dostępu: 15 marzec 2019].

[8] D. G. Lowe, „Object Recognition from Local Scale-Invariant Features," University of British Columbia, Vancouver, B.C., Canada, 1999.

[9] E. Lee, S.-S. Lee, Y. Hwang i S.-J. Jang, „Hardware implementation of fast traffic sign recognition for intelligent vehicle system,” w *2016 International SoC Design Conference (ISOCC)*, Jeju, South Korea, 23-26 październik 2016.

[10] D. Deguchi, M. Shirasuna, K. Doman, I. Ide i H. Murase, „Intelligent traffic sign detector: Adaptive learning based on online gathering of training samples,” w *2011 IEEE Intelligent Vehicles Symposium (IV)*, Baden-Baden, Germany, 05 lipiec 2011.

[11] S. Maldonado-Bascon, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno i F. Lopez-Ferreras, „Road-Sign Detection and Recognition Based on Support Vector Machines,” *IEEE Transactions on Intelligent Transportation Systems*, tom 8, nr 2, pp. 264 - 278, 04 czerwiec 2007.

[12] H. Liu, D. Liu i J. Xin, „Real-time recognition of road traffic sign in motion image based on genetic algorithm,” w *Proceedings. International Conference on Machine Learning and Cybernetics*, Beijing, China, 4-5 grudzień 2002.

[13] A. d. l. Escalera, J. M. Armingol i M. Mata, „Traffic sign recognition and analysis for intelligent vehicles,” *Image and Vision Computing*, tom 21, nr 3, pp. 247-258, marzec 2003.

[14] P. Sermanet i Y. LeCun, „Traffic sign recognition with multi-scale Convolutional Networks,” w *The 2011 International Joint Conference on Neural Networks*, San Jose, CA, USA, 21 lipiec 2011.

[15] Y. Freund i R. E. Schapire, „Experiments with a New Boosting Algorithm,” w *Proceedings of the Thirteenth International Conference*, New Jersey, 1996.

[16] Y. Yang, H. Luo, H. Xu i F. Wu, „Towards Real-Time Traffic Sign Detection and Classification,” *IEEE Transactions on Intelligent Transportation Systems*, tom 17, nr 7, pp. 2022 - 2031, czerwiec 2016.

[17] A. d. l. Escalera, L. E. Moreno, M. A. Salichs i M. Armingol, „Road Traffic Sign Detection and Classification,” *IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS*, tom 44, nr 6, pp. 848 - 859, grudzień 1997.

[18] R. Duda i P. Hart, „Use of the Hough Transformation To Detect Lines and Pictures,” Stanford Research Institute, Menlo Park, California, 1972.

[19] G. Wang, G. Ren, Z. Wu, Y. Zhao i L. Jiang, „A hierarchical method for traffic sign classification with support vector machines,” w *The 2013 International Joint Conference on Neural Networks (IJCNN)*, Dallas, TX, USA, 4-9 sierpień 2013.

[20] I. M. Creusen, R. G. Wijnhoven, E. Herbschleb i P. D. With, „Color exploitation in hog-based traffic sign detection,” w *International Conference on Image Processing*, Hong Kong, 26-29 wrzesień 2010.

[21] P. Sermanet, D. Eigen, Z. Xiang, M. Michael, F. Rob i L. Yann, „Integrated recognition localization and detection using convolutional networks," Courant Institute of Mathematical Sciences, New York, NY, USA, 21 grudzień 2013.

[22] Z. Huang, Y. Yu, J. Gu i H. Liu, „An Efficient Method for Traffic Sign Recognition Based on Extreme Learning Machine," *IEEE Transactions on Cybernetics,* tom 47, nr 4, pp. 920 - 933, 14 marzec 2016.

[23] T. Ojala, M. Pietikäinen i D. Harwood, „Performance evaluation of texture measures with classification based on Kullback discrimination of distributions," w *Proceedings of 12th International Conference on Pattern Recognition*, Jerusalem, Israel, Israel, 9-13 październik 1994.

[24] P. Viola i M. Jones, „Robust Real-time Object Detection," w *Second International Workshop on Statistical and Computational Theories of Vision*, Vancouver, Canada, 13 sierpień 2001.

[25] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer i T. Koehler, „A system for traffic sign detection, tracking, and recognition using color, shape, and motion information," w *IEEE Proceedings. Intelligent Vehicles Symposium*, Las Vegas, NV, USA, USA, 6-8 czerwiec 2005.

[26] N. Barnes i A. Zelinsky, „Real-time radial symmetry for speed sign detection," w *IEEE Intelligent Vehicles Symposium*, Parma, Italy, Italy, 14-17 czerwiec 2004.

[27] G. Loy i N. Barnes, „Fast shape-based road sign detection for a driver assistance system," w *International Conference on Intelligent Robots and Systems*, Sendai, Japan, 28 wrzesień 2004.

[28] T. Szymczyk, „Metoda dopasowania wzorców w rozpoznawaniu obrazów – ograniczenia, problemy i modyfikacje," w *Automatyka*, Lublin, Wydział Elektrotechniki i Informatyki, Politechnika Lubelska, 2008, pp. 449 - 462.

[29] C.-Y. Fang, S.-W. Chen i C.-S. Fuh, „Road sign detection and tracking," *Transactions on Vehicular Technology*, tom 52, nr 5, pp. 1329 - 1341, 23 wrzesień 2003.

[30] P. S. Miguel i A. R. Alastair, „Using self-organising maps in the detection and recognition of road signs," *Image and Vision Computing,* tom 6, nr 27, pp. 673-683, maj 2009.

[31] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li i S. Hu, „Traffic-Sign Detection and Classification in the Wild," w *2016 IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 27-30 czerwiec 2016.

[32] F. Zaklouta, B. Stanciulescu i O. Hamdoun, „Traffic sign classification using K-d trees and random forests," w *Proceedings of International Joint Conference of Neural Networks*, San Jose, California, USA, 31 lipiec 2011.

[33] M. Mathias, R. Timofte, R. Benenson i L. V. Gool, „Traffic sign recognition-how far are we from the solution?," w *The 2013 International Joint Conference on Neural Networks (IJCNN)*, Dallas, TX, USA, 4-9 sierpień 2013.

[34] P. Dollár, Z. Tu, P. Perona i S. Belongie, „Integral Channel Features," Dept. of Electrical Engineering, Los Angeles, Ca, USA, 2009.

[35] R. Kastner, T. Michalke, T. Burbach, J. Fritsch i C. Goerick, „Attention-based traffic sign recognition with an array of weak classifiers," w *2010 IEEE Intelligent Vehicles Symposium*, San Diego, CA, USA, 21-24 czerwiec 2010.