

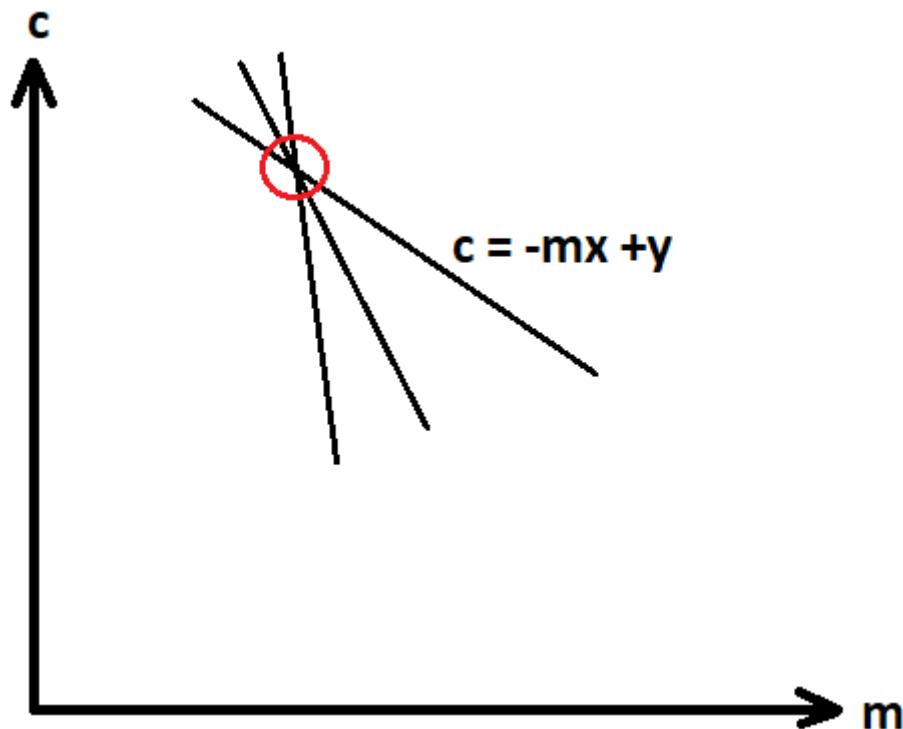
Sterowanie pojazdem może odbywać się poprzez użycie kontrolera wykorzystującego fale radiowe do wydawania poleceń docierających do odbiornika znajdującego się w pojeździe, które mikro-kontroler przekształca na instrukcje mechaniczne, ale również poprzez program znajdujący się na płycie scalonej w pojeździe tworząc w ten sposób pojazd autonomiczny.

Oprogramowanie pojazdów autonomicznych może wykorzystać do ich sterowania między innymi analizę otoczenia przez wzory matematyczne albo algorytmy sztucznej inteligencji. Poza logiką poruszania się pojazdów, w większości prototypowych lub wyspecjalizowanych, na przykład poruszających się w fabrykach po określonej trasie, należy zadbać o adaptację jazdy do zmieniających warunków na drodze, aby zapewnić bezpieczeństwo pojazdu oraz przedmiotów i ludzi w przestrzeni otaczającej pojazd.

Roboty podążające za linią poprzez jej wykrywanie czujnikami podczerwonymi, a także zliczanie pikseli w danym kolorze z dwóch stron obrazu, są użyteczne tylko w kontrolowanym środowisku, jednakowoż ukazują ideę, za którą pojazdy poruszające się po drogach mogą podążyć. Oparcie się na podążaniu za namalowaną linią powinno zostać zastąpione wykrywaniem linii, aby pojazd mógł wykorzystać tę ideę na każdej drodze, a także do wykrywania otoczenia w okolicy jezdni pozwalając dostosować się do przepisów ruchu drogowego.

W takim razie należałoby wykryć linię poprzez algorytmy analizy obrazu. Obraz do analizy powinno się zmienić na obraz krawędzi. Krawędź jest to linia powstała poprzez znaczącą oraz gwałtowną zmianę w jasności pikseli w obrazie. Wykrywanie krawędzi odbywa się poprzez obliczenie gradientów wartości jasności pikseli w częściach obrazu, powstałych poprzez podzielenie obrazu na kwadraty o wybranej długości boku liczonej w pikselach, a następnie znalezienie miejsc, w których linia została wykryta najczęściej z uwzględnieniem kierunku gradientu. W finalnym obrazie uwzględniane są tylko znalezione wartości maksymalne. Analiza samych krawędzi pozwala zredukować liczbę pikseli poddawanych transformacji. Podczas wykrywania linii należy zastosować wersję obrazu w odcieniach szarości, aby obliczenia gradientu odbywało się w jednym wymiarze barw. [1]

Algorytmem pozwalającym wykrywać linie w niedoskonałym obrazie jest Transformacja Hough'a [2]. W tej transformacji dla każdego punktu (x,y) wybierane są parametry m oraz c prostej, przechodzącej przez ten punkt, gdzie parametry m i c znajdują się we wzorze na prostą $y = mx + c$. Wzór jest przekształcany na postać $c = -mx + y$, a więc teraz należy traktować wartości x oraz y badanego punktu jako parametry prostej, analogicznie do parametrów m i c w podstawowym wzorze. Ten układ współrzędnych jest nazywany przestrzenią Hough'a. Linia w przestrzeni Hough'a przedstawia wszystkie wartości parametrów m i c , prostych przechodzących przez punkt (x,y) w badanym obrazie. Rysowanie linii jest powtarzane dla każdego z punktów w obrazie. W ten sposób powstają linie dla każdego punktu, reprezentujące parametry prostych przechodzących dany punkt, które przecinają się w miejscach, gdzie punkty (x,y) w badanym obrazie mają wspólną prostą. Prosta ta ma parametry m oraz c możliwe do odczytania z osi w przestrzeni Hough'a. Wybierane są tylko punkty (m,c) , w których przecina się największa liczba prostych co oznacza, że została wykryta linia o danych parametrach. [2]

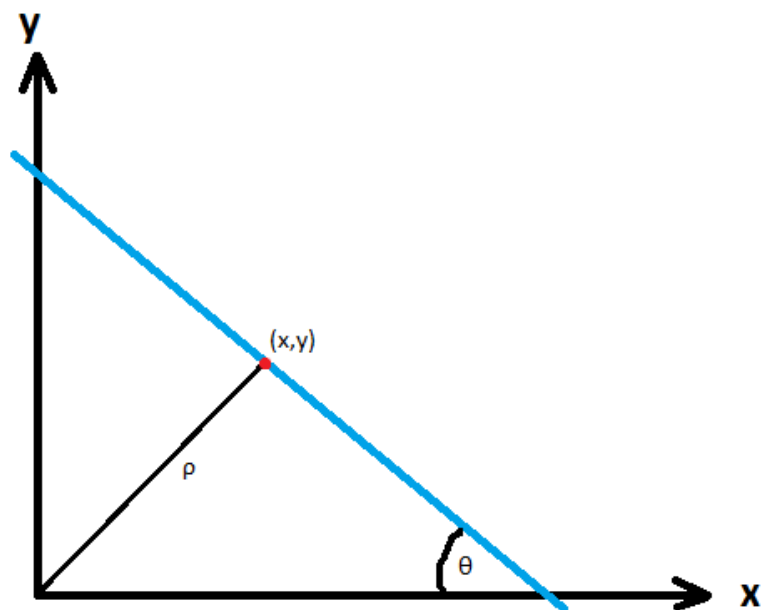


Rys 1. Proste przecinające się w przestrzeni Hough'a

W programowaniu ten algorytm może zostać zaimplementowany poprzez stworzenie tablicy zer, a następnie inkrementowanie wartości w miejscach gdzie linia z parametrami m oraz c przechodzi przez punkt. Na koniec wybierane są komórki z największą liczbą i odpowiadające im wartości parametrów m oraz c są parametrami wykrytych prostych.

Pierwotne podejście do obliczania parametrów daje możliwość przecięcia się prostych w przestrzeni Hough'a w miejscu bardzo oddalonym od środka układu współrzędnych. Stwarza to problem w podejściu programistycznym jako, że tablica ma swój rozmiar, a więc nie jest w stanie pomieścić wszystkich wartości lub będzie niepraktyczna przez swój rozmiar zajmujący dużą ilość pamięci. Zawsze jest możliwość obliczenia miejsca przecięcia się prostych ze wzoru na punkt przecięcia się dwóch prostych ale wymaga to dodatkowych zasobów komputera.

Prostszą metodę zaproponowali Richard Duda oraz Peter Hart. Metoda ta polega na wykorzystaniu wzoru na prostą w postaci $p = x \cdot \cos\theta + y \cdot \sin\theta$. Parametrami są p , odległość prostej od środka układu współrzędnych, oraz θ , kąt jaki prosta tworzy z osią x . Wartości wybierane są przez program o pewien ustalony krok od zera do 180° dla kąta θ oraz od 0 do długości przekątnej obrazu dla odległości p . Dla tych parametrów w przestrzeni Hough'a z osiami θ oraz p rysowane są sinusoidy dla wartości x oraz y . Podobnie jak w pierwotnym podejściu, tutaj również miejsca przecięcia się tych sinusoid reprezentują parametry prostych. [2]



Rys 2. Prosta z parametrami ρ oraz θ

Ta metoda pozwala na wykrycie kształtu drogi, po której pojazd porusza się. Linie kształtu drogi powinny zbiegać się w górnej części obrazu oraz nierówności nie powinny mieć na nią wpływu, a więc należy ograniczyć szumy poprzez wybieranie prostych wykrytych tylko dla największej liczby punktów. Dalsze ograniczenie zużycia mocy obliczeniowej, które jest niezbędne w analizie obrazu w czasie rzeczywistym, może zostać zapewnione poprzez zdefiniowanie regionu zainteresowania obejmującego tylko część drogi z przodu pojazdu kończącego się przed miejscem zbiegu dwóch krawędzi jezdni.[3]

Algorytm wykrywania drogi powinien również wykrywać zakręty poprzez analizę zmiany kierunku prostej względem już wykrytej. Jeżeli wykrywana prosta zmienia kierunek względem prostej wykrytej wcześniej, pojazd powinien rozpoznać zakręt lub skrzyżowanie i zachować się w zaprogramowany sposób dla tych formacji drogowych[3]. Poza wykryciem drogi należy wykrywać inne pojazdy, co można osiągnąć przez ultradźwiękowe czujniki odległości lub LiDAR'y(ang. *Light Detection and Ranging*), tak aby nie zwiększać obciążenia maszyny dodatkowymi obliczeniami.

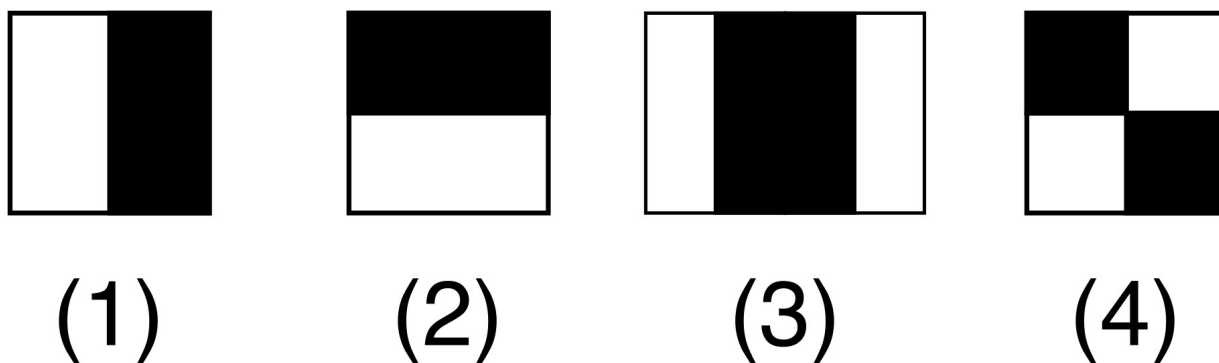
Światła jak i znaki drogowe mogą zostać wykryte przez Ogólną Transformatę Hough'a, która może wykryć każdy obiekt, który można przedstawić wzorem w układzie kartezjańskim. Kołowa Transformata Hough'a będzie korzystać z trzech zmiennych wartości r – długości promienia, a – współrzędnej środka koła na osi poziomej, b – współrzędnej środka koła na osi pionowej, gdzie dla każdej wykrytej figury obraz w jej zakresie będzie klasyfikowany przez sztuczną inteligencję w porównaniu do wzorców znaków i świateł drogowych [4]. Schemat jest podobny do wykrywania linii, tylko w tym przypadku tablica lub układ współrzędnych będą posiadały trzy wymiary, a także w przestrzeni Hough'a będą rysowane koła, których przecięcia będą reprezentować wartości parametrów środka szukanych kół. Trzeci wymiar może zostać pominięty jeżeli znamy promień szukanego koła [4].

Zamiast analizy linii mogą zostać użyte algorytmy uczenia maszynowego na wybranym fragmencie obrazu, w którym oczekujemy znaków drogowych. Jest również możliwość wykrywania znaków drogowych, a następnie ich klasyfikacja poprzez algorytmy sztucznej inteligencji[5].

Jednym z algorytmów uczenia maszynowego pozwalających na rozpoznawanie obra-

zów jest Algorytm Lokalnych Binarnych Wzorów, który dzieli obraz na części i porównuje wartości jasności pikseli na około środkowego, zapisując 0 jak jest mniejszy i 1 jak jest większy tym samym tworząc liczbę binarną, która w połączeniu z pozostałymi liczbami zliczonymi w pozostałych częściach obrazu, o rozmiarach wybranych przy implementacji, jest porównywana z liczbami wyliczonymi dla obrazów podczas nauki algorytmu. Algorytm ten wymaga aby obraz był w odcieniach szarości, aby porównywanie odbywało się w jednym wymiarze barw[6].

Innym algorytmem, realizującym analizę obrazu, podanym w [5] jest Algorytm Haar, który działa na podobnej zasadzie, a więc również potrzebuje obrazu w odcieniach szarości oraz podzielenia obrazu na mniejsze części. Jednak zamiast zliczać piksele tworząc liczbę binarną, działa na zasadzie sprawdzania czy w danej części obrazu znajduje się własność Haar poprzez obliczanie średniej arytmetycznej wartości jasności pikseli w miejscach, gdzie własność Haar powinna mieć ciemne piksele, a następnie odejmując od wyniku średnią jasność jasnych pikseli, obliczoną w miejscu, gdzie powinny się znajdować. Wartości są wysokie dla ciemnych pikseli i niskie dla jasnych pikseli. Czym bardziej różnica jest bliższa wartości 1 tym jest większa prawdopodobieństwo wystąpienia w tym miejscu danej własności. Własność Haar jest wzorcem jak wygląda część obrazu, do której algorytm próbuje dopasować części analizowanego obrazu. Wszystkie wykryte własności Haar są zliczane i porównywane z własnościami obrazu zliczonymi podczas nauki algorytmu.[7].



Rys 3. Przykłady własności Haar [7]
 źródło: https://en.wikipedia.org/wiki/Haar-like_feature

Cytowana Literatura:

- [1] Socret Lee <https://towardsdatascience.com/lines-detection-with-hough-transform-84020b3b1549>
- [2] https://en.wikipedia.org/wiki/Hough_transform
- [3] PANNU, Gurjashan Singh; ANSARI, Mohammad Dawud; GUPTA, Pritha. Design and implementation of autonomous car using Raspberry Pi. International Journal of Computer Applications, 2015, 113.9.
- [4] Nath, Jewel & Rajee, Alimul. (2022). Multi-functional Autonomous Car using raspberry pi and python.. https://www.researchgate.net/publication/367167563_Multi-functional_Autonomous_Car_using_raspberry_pi_and_python
- [5] Sirbu, Maria-Adelina & Baiasu, Andrei & Bogdan, Razvan & Crişan-Vida, Mihaela. (2018). Smart traffic sign detection on autonomous car. 1-4. 10.1109/ISCTC.2018.8583948.

https://www.researchgate.net/publication/329840135_Smart_traffic_sign_detection_on_autonomous_car

[6]https://en.wikipedia.org/wiki/Local_binary_patterns

[7]https://en.wikipedia.org/wiki/Haar-like_feature