analiza scen, detekcja obiektów, obrazy cyfrowe, rozpoznawanie znaków

Włodzimierz Kasprzak *, Marcin Jankowski

Implementacja czujnika wizyjnego w systemie sterowania ruchem drogowym

1 Sterowanie ruchem ulicznym w projekcie OMNI

Projekt badawczy Unii Europejskiej o nazwie OMNI ("Open Model For Network-wide Heterogeneous Intersection-based Transport Management") [1] wprowadza model oprogramowania o otwartej architekturze przeznaczony do zarządzania ruchem drogowym w mieście. W modelu wyróżniono: centralny serwer zwany OMNI-MOUN oraz standardy interfejsów dla aplikacji programowych i urządzeń w polu. Celem standaryzacji była integracja technologii (aplikacje nadzorujące i sterujące ruchem, inteligentne czujniki, usługi informacyjne WWW, itp.) pochodzącej od różnych producentów.

Aplikacja w OMNI, nadzorująca ruch uliczny, ma za zadanie automatyczną detekcję wypadków lub zakłóceń ruchu (Rys. 1). Aplikacja sterująca ruchem oznacza w tym przypadku połączenie dwóch wzajemnie uzupełniających się trybów sterowania przełączaniem świateł na skrzyżowaniach ulic: pierwszy tryb oznacza lokalną pracę sterownika na skrzyżowaniu (czyli urządzenia polowego pracującego w czasie rzeczywistym), drugi jest realizowany przez aplikacje programowe przeznaczone do śledzenia natężenia ruchu i realizacji strategii przełączania świateł na poziomie całej sieci ulic w aglomeracji.

Kolejną ważną aplikacją OMNI jest dostarczanie indywidualnym użytkownikom systemu aktualnej informacji o sytuacji na ulicach miasta poprzez Internet i usługi WWW. Usługi takie mogą obejmować prezentację sytuacji na wybranej drodze, planowanie przejazdu w trybie rzeczywistym, przedstawianie alternatywnych tras i dostarczanie do zarejestrowanych użytkowników informacji zgodnej z wybranym przez nich profilem.

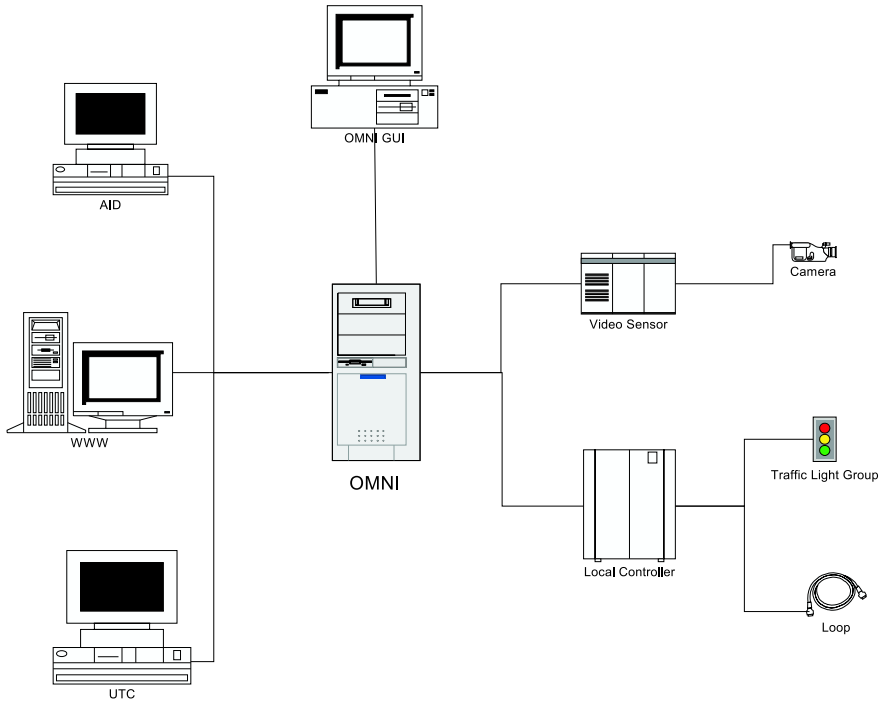
Kolejny rodzaj aplikacji OMNI stanowi zarządzanie korporacyjnym parkiem pojazdów, z wykorzystaniem urządzeń GPS/GSM i czujników rozmieszczonych na drogach, którego celem jest lokalizacja pojazdów i optymalizacja tras przejazdów.

Z punktu widzenia oprogramowania model OMNI zapewnia definicje klas i interfejsów, realizujących programowy model klient-serwer, przeznaczonych do opisu urządzeń i sytuacji występujących w ruchu drogowym i na skrzyżowaniach ulic (Rys. 2). Jednym z urządzeń w modelu OMNI jest inteligentny czujnik wizyjny. Jego zadaniem jest akwizycja i analiza obrazu w trybie rzeczywistym [2], [3] (Rys. 3).

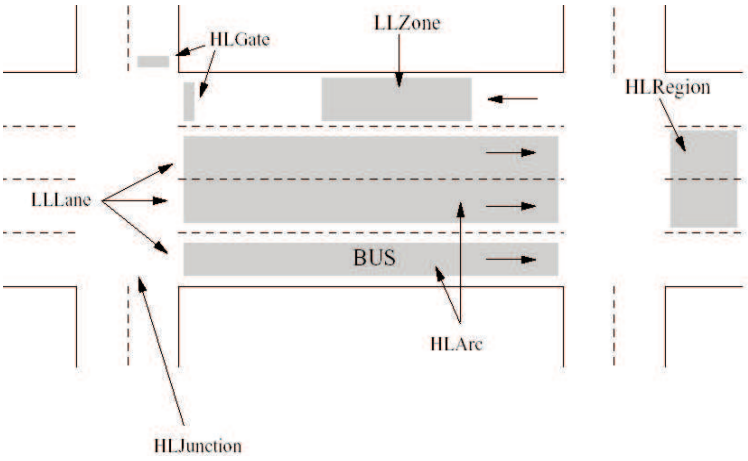
W artykule przedstawiono szczegóły implementacyjne (program) modularnego czujnika wizyjnego w ramach OMNI, którego ogólny projekt został opisany we wcześniejszym artykule [9]. Analiza obrazu została rozdzielona na następujące moduły programowe [4]: (1) auto-

*Instytut Automatyki i Informatyki Stosowanej, Politechnika Warszawska, ul. Nowowiejska 15/19, 00-665 Warszawa. E-mail: W.Kasprzak@ia.pw.edu.pl , M.Jankowski@elka.pw.edu.pl

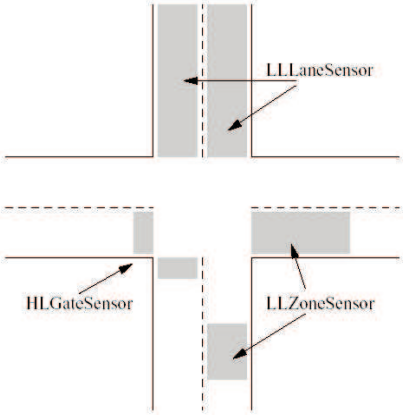
kalibracja kamery, (2) detekcja natężenia ruchu, (3) detekcja zajętości pasa ruchu (kolejki pojazdów), (4) rozpoznawanie tablic rejestracyjnych. Uzupełnieniem programu jest moduł komunikacji z serwerem OMNI-MOUN, zarządzającym bazą danych o sytuacji w mieście. Z punktu widzenia programisty każdy moduł wizyjny jest realizowany przez pojedynczą klasę co umożliwia ewentualne łatwe rozdzielenie programu na szereg odrębnych procesów (np. obiektów bibliotek DCOM lub CORBA), jeśli wymagana jest ich rozproszona implementacja.



Rysunek 1. Przykład architektury systemu informacyjnego według modelu OMNI.



Rysunek 2. Pojęcia związane z ruchem drogowym, przewidziane w implementacji serwera OMNI-MOUN [1].



Rysunek 3. Rozmieszczenie urządzeń na skrzyżowaniu i odpowiadające im pojęcia w OMNI-MOUN [1].

2 Komunikacja z OMNI-MOUN

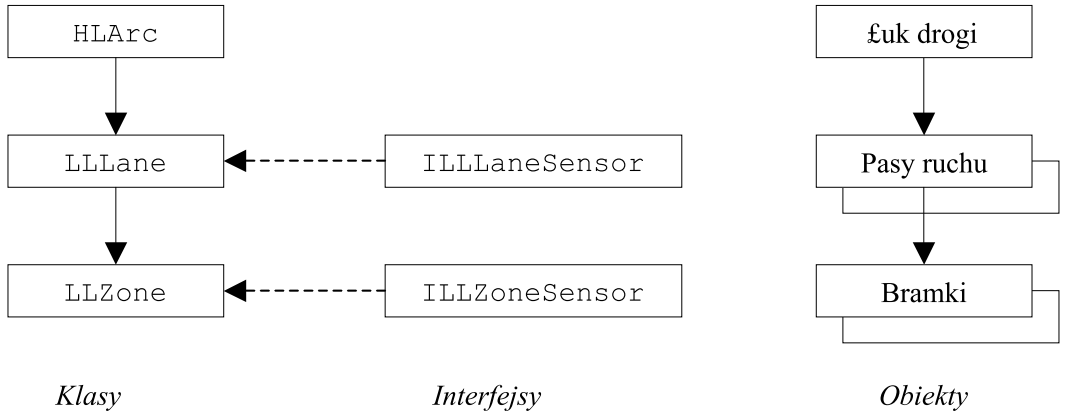
Podczas inicjalizacji serwera OMNI tworzone są obiekty odpowiadające fizycznie istniejącym skrzyżowaniom i pasom ruchu.

Nasze urządzenie, a konkretnie jego moduł komunikacji, stanowi obiekt DCOM będący klientem serwera OMNI-MOUN). Podczas inicjalizacji programu, klient pobiera wskaźnik do interfejsu obiektu serwera, zarządzającego obiektami w OMNI-MOUN - tzw. `MOUNManager`. Następnie rejestrujemy nasze urządzenie wywołując metodę `IMOUNManager::getIdsOfType()` i przekazując w niej typ czujnika - klasę `LLFieldVideoSensor`. Odtąd naszemu urządzeniu odpowiadać będzie nowo utworzony obiekt klasy `LLFieldVideoSensor`.

W dalszej kolejności program pobiera z OMNI-MOUN wskaźniki do obiektów, właściwych dla rozmieszczenia urządzenia w polu, w następującej kolejności (Rys. 4):

- obiekt "łuku drogi" pomiędzy skrzyżowaniami (obiekt klasy `HLArc`), wywołując metodę `IHLArcContainer::getSensorsOnArc()`,
- obiekty dla pasów ruchu związanych z pobranym obiektem typu `HLArc` (obiekty klasy `LLLane`), wywołując `ILLLaneContainer::getSensorsOnLane()`,
- obiekty "bramek" przeznaczone do detekcji natężenia ruchu (typu `LLZone`, powiązane z obiektami typu `LLLane`), wywołując `ILLZoneContainer::getSensorsOnZone()`.

Procedury wymiany informacji pomiędzy OMNI-MOUN a naszym programem czujnika wizyjnego wyznaczone są przez interfejsy DCOM, zdefiniowane w OMNI: `ILLFieldVideoSensor` - dla sterowania procesem analizy obrazu z MOUN, `ILLLaneSensor` - dla przekazywania przez czujnik informacji o kolejce pojazdów (zajętości pasa ruchu), `ILLZoneSensor` - dla informowania o natężeniu ruchu. W szczególności wyniki analizy obrazu przekazywane są przez cykliczne wywołania metody `Measure`, zadeklarowanej w obu interfejsach: `ILLZoneSensor` i `ILLLaneSensor`.

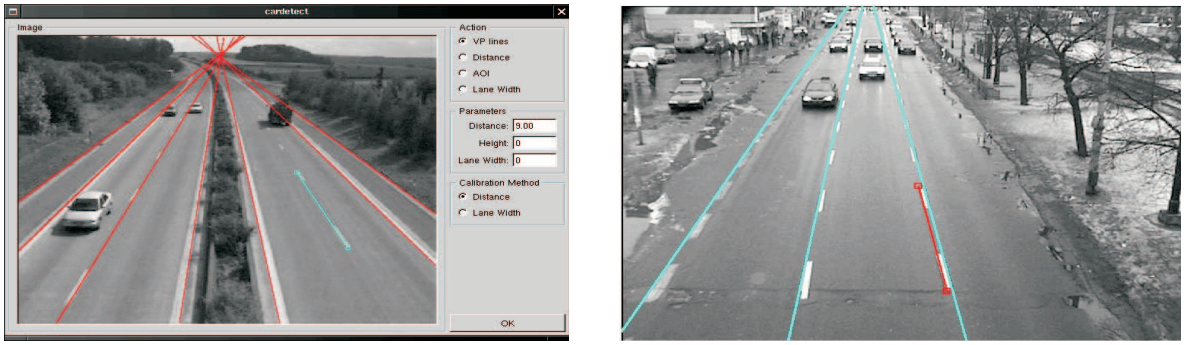


Rysunek 4. Struktura klas i interfejsów wykorzystywanych w komunikacji programu czujnika wizyjnego z OMNI-MOUN.

3 Moduł auto-kalibracji

Przy założeniu prostego modelu kamery - rzutu zbieżnego bez zniekształceń - wymagana jest znajomość 6 parametrów geometrycznych: położenia środka układu kamery (X_c, Y_c, Z_c) i zorientowania osi kamery w układzie odniesienia (α, β, γ); a także 4 parametrów wewnętrznych: długości ogniskowej (F), rozmiaru piksela s_P) i położenia środka obrazu w układzie kamery (X_0, Y_0)).

Ustalenie parametrów wewnętrznych odbywa się poza systemem (za wyjątkiem ogniskowej, która może być zmieniana podczas pracy urządzenia), natomiast ustalenie parametrów geometrycznych jest celem modułu auto-kalibracji. Przyjęto w nim założenia upraszczające: wysokość środka układu kamery nad płaszczyzną drogi jest znana (tzn. $Y_c = H$), a parametr Z_c może być ustalony a priori (np. $Z_c = 0$). W ten sposób pozostają jeszcze do ustalenia 4 parametry: (α, β, γ) i X_c (położenie kamery względem środkowej osi drogi).



Rysunek 5. Kalibracja kamery: (lewa strona) okno dialogowe użytkownika przeznaczone do kalibracji kamery; (prawa strona) przykład procesu kalibracji (niebieskie linie są wykrywane automatycznie, czerwona linia jest zaznaczana przez użytkownika i mierzona w terenie (ustalanie skali).

W pracy zaimplementowana została procedura opisana w [5]. Użytkownik powinien zaznaczyć na obrazie odcinek, którego rzeczywistą długość w terenie uprzednio zmierzył i przekazał tę długość programowi, podobnie jak informację o 4 parametrach wewnętrznych i wysokości H. Program wykonuje automatycznie detekcję linii prowadzących do punktu zbieżności rzutowania (VP) (Rys. 5), co pozwala na ustalenie pozostałych parametrów geometrycznych.

4 Detekcja natężenia ruchu i długości kolejki

Podany moduł programu realizuje nadążną analizę sekwencji obrazów zwracając pomiary dla każdego pasa ruchu, wybranego przez użytkownika, dotyczące:

- *natężenia ruchu*, mierzonego liczbą pojazdów na minutę, przekraczających zaznaczoną w obrazie bramkę (Rys. 6);
- *długości kolejki* (alternatywnie - zajętości pasa ruchu) - "mierząc" rzeczywistą długość szeregu pojazdów wykrytych w obrazie (Rys. 7).

Użytkownik zaznacza na obrazie odcinek, wyznaczający miejsce zliczania pojazdów (Rys. 6). Na bieżąco odbywa się analiza histogramu liczonego dla tego odcinka obrazu. Dla pasa

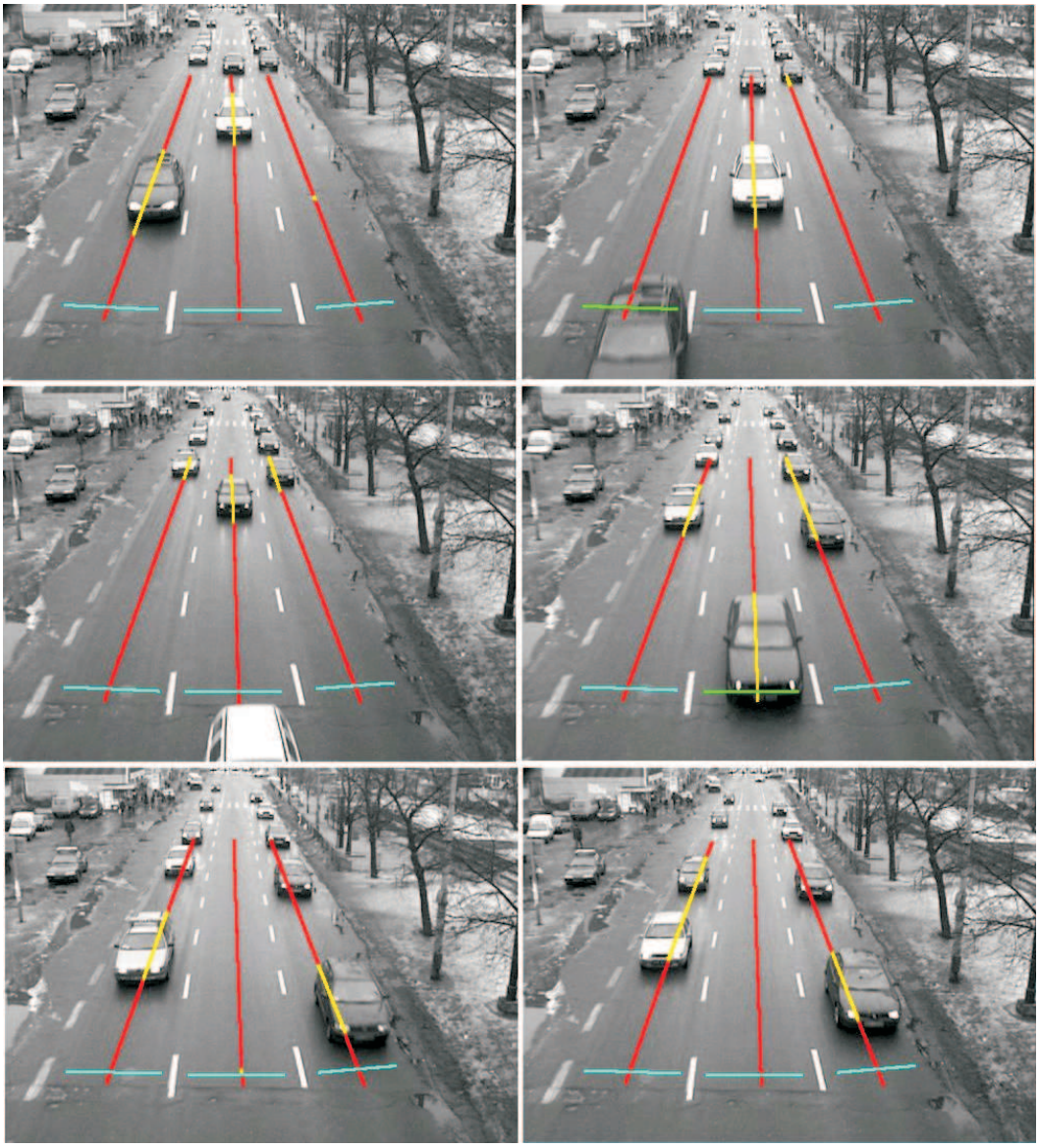
Rysunek 6. Linie poziome wyznaczają bramki, w których zliczana jest liczba pojazdów. Linie pionowe reprezentują pasy ruchu, wzdłuż których mierzona jest długość kolejki.

Rysunek 7. Przykład detekcji kolejki pojazdów: pojazd powoduje powstanie specyficznego histogramu jasności liczonego dla wiersza obrazu - złożonego z 2 obszarów "jasnych" i 2 obszarów "ciemnych".

Rysunek 8. Kolejki wykrywane są oddzielnie dla każdego zaznaczonego pasa ruchu.

ruchu z widocznym pojazdem taki histogram różni się w istotny sposób od histogramu "spoczynkowego" (tzn. braku pojazdu) - występuje w nim przewaga obszarów "jaśniejszych" od tła lub "ciemniejszych" od tła. Pierwsze "przełączenie" histogramu dla zadanego odcinka powoduje zwiększenie licznika pojazdów i jest ona pamiętana do chwili powrotu do histogramu o "normalnym" wyglądzie charakteryzującym tło.

Detekcja kolejki (lub stopnia zajętości pasa jezdni) wymaga detekcji na raz wszystkich pojazdów dla danego pasa ruchu. Badany pas wyznaczany jest przez użytkownika, który kreśli na obrazie odcinek przebiegający wzdłuż rzutu osi danego pasa ruchu (Rys. 7). Przy "pustym" pasie następuje rejestracja zakresu jasności, charakteryzującego jezdnię. Podczas aktywnej pracy analizowany jest rozkład jasności obrazu wzdłuż wskazanego odcinka - pod-odcinki złożone z pikseli jaśniejszych lub ciemniejszych od tła wyznaczają nam zajętość pasa ruchu przez pojazd. Rzeczywista długość kolejki w terenie obliczona zostaje po wstecznym rzutowaniu na płaszczyznę ulicy odcinków obrazu odpowiadających wykrytym pojazdom (Rys. 8). Alternatywnie można na tej podstawie podać stopień zajętości badanego pasa jezdni przez pojazdy.



Rysunek 9. Przykłady zliczania pojazdów (detekcja pojazdu sygnalizowana jest zmianą koloru linii z niebieskiego na zielony) i detekcji kolejki (zajetość jezdni przez pojazd sygnalizowana jest zmianą czerwonej linii na zieloną).

Przeprowadzono testy pracy modułów detekcji natężenia ruchu i detekcji długości kolejki, w trybie *off-line*, na sekwencjach obrazów przedstawiających ruch uliczny w Warszawie [4] (Rys. 9). Określono i przeanalizowano czasy pracy poszczególnych modułów na dostępnym procesorze o wydajności 345.5 Mflop (Tab. 1).

5 Rozpoznawanie tablic rejestracyjnych

Moduł rozpoznawania tablic rejestracyjnych wykonuje trzy zasadnicze zadania: (A) detekcję prostokątnego obszaru w obrazie, w którym może wystąpić tablica rejestracyjna pojazdu; (B) detekcję obszarów dla poszczególnych znaków wewnątrz obszaru tablicy ([4], [6]); (C) klasyfikacja znaków (może tu znaleźć zastosowanie komercyjny pakiet OCR) lub nasza własna prosta i szybka procedura oparta na dopasowaniu pomiaru ze wzorcową bitmapą ([7], [8]).

Moduł	Czas analizy (1000 ramek)	Wymagane MFlopy (przy 25 ramkach/sekundę)
Długość kolejki	1.77 s	15.3 MFlop
Natężenie ruchu	0.37 s	3.2 MFlop
Rozpoznawanie tablic	14.6 s	126 MFlop

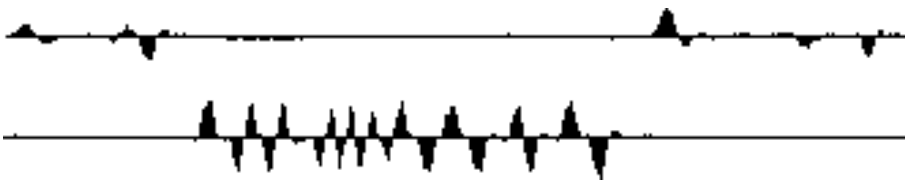
Tabela 1. Analiza szybkości pracy trzech modułów czujnika wizyjnego.

Rysunek 10. Idea etapu detekcji obszaru tablicy - analiza rozkładów jasności wzdłuż linii obrazu.

Pierwsze zadanie może być zrealizowane poprzez detekcję gradientów funkcji jasności wzdłuż wierszy obrazu (Rys. 10). Znalezienie kilku kolejnych wierszy przejawiających dużą zmienność tego gradientu prowadzi do generacji hipotezy o obszarze zawierającym tablicę rejestracyjną. "Duża zmienność" wystąpi wtedy, gdy po kolei występują wystarczająca liczba par gradientów o dużych amplitudach ale przeciwnych zwrotach (Rys. 11).

Dla każdej hipotezy obszaru tablicy wykonywane jest drugie zadanie modułu - dekompozycja obszaru na obszary dla poszczególnych znaków. W realizacji tego zadania wyróżnimy:

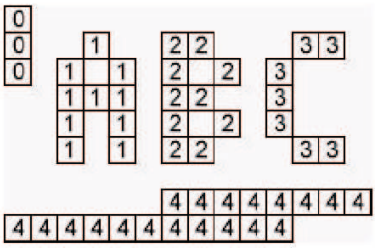
1. Usunięcie składowej stałej jasności.
2. Obliczenie obrazu binarnego metodą progowania.
3. Detekcja spójnych łańcuchów pikseli w obrazie binarnym. Dla każdego łańcucha określamy jego rozmiary, obszar i położenie środka masy (Rys. 12).
4. Filtracja rozmiarów - usunięcie łańcuchów zbyt małych i zbyt dużych na to, aby reprezentowały znak. Progi dobierane są względem rozmiaru obszaru tablicy.



Rysunek 11. Dyskretne 1-sze pochodne funkcji jasności wzdłuż 2 wierszy obrazu.

5. Detekcja obszarów zawierających pojedyncze znaki. Podstawą tego kroku są 2 histogramy obszaru liczone wzdłuż osi X i Y (Rys. 13).

6. Ostatecznie w obrazie binarnym obszaru tablicy generowane są jednorodnie rozłożone prostokątne obszary, zawierające pojedyncze znaki (Rys. 14).



Rysunek 12. Detekcja i etykietowanie łańcuchów pikseli w obrazie binarnym.

Rysunek 13. Analiza 2 histogramów - wzdłuż osi X i Y - w celu detekcji pojedynczych znaków.

Rysunek 14. Przykład znalezionych prostokątów dla pojedynczych znaków.

Okno dialogowe "detekcja tablicy"

W tym oknie dialogowym po jego prawej stronie pokazywany jest analizowany obraz (Rys. 15). Użytkownik może ustawić aktualne parametry analizy "ręcznie" lub poprzez zaznaczenie obszaru znaków tablicy w obrazie a następnie wciśnięcie przycisku "Calibrate" - parametry zostaną dobrane automatycznie.

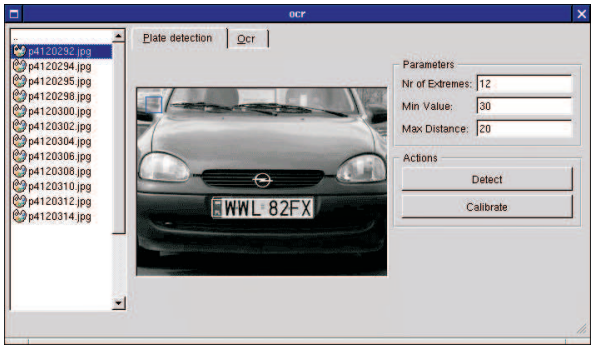
Okno dialogowe "OCR"

Zrealizowano procedurę do klasyfikacji znaków tablicy zgodnej z polskim standardem. Użytkownik ma możliwość nauczenia klasyfikatora swoimi własnymi znakami. Na rysunku 16 pokazano, przeznaczone do tego kroku, okno dialogowe. Zawiera ono: powiększony obraz obszaru tablicy, powiększenie wybranego znaku tablicy, przyciski i elementy menu. Użytkownik ma możliwość interakcyjnej pracy z systemem w celu wyboru znaków z obrazu, opisaniu ich i dodaniu do bazy znaków.

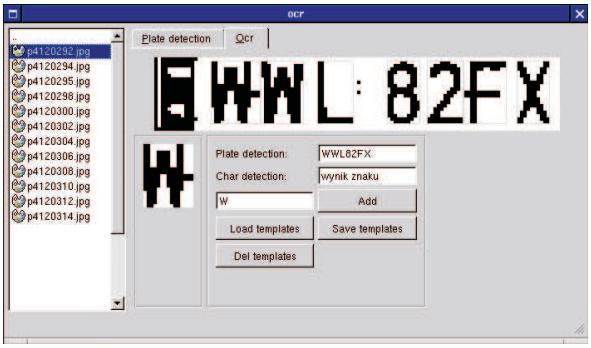
Istnieje szereg podejść do klasyfikacji znaków. Zobaczmy zasady dwóch najprostszych metod (Rys. 17): (A) pomiar kilku odległości znaku do linii bazowej dolnej i górnej, (B) zliczanie liczby przecięć znaku z kilkoma prostymi poziomymi i pionowymi. Zrealizowano podejście polegające na dopasowaniu obszaru znaku z modelem w postaci bit-mapy o znormalizowanym rozmiarze (Rys. 18).

6 Podsumowanie

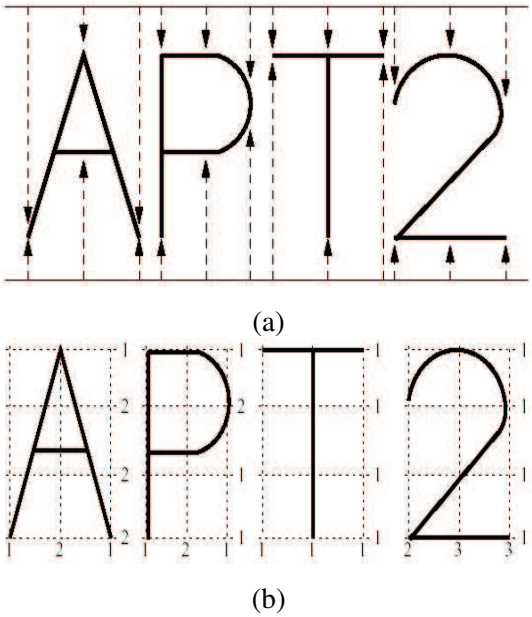
Przedstawiono programową implementację wizyjnego czujnika ruchu drogowego i jego współpracę w charakterze klienta w systemie OMNI - integrującego aplikacje informacyjno-sterujące,



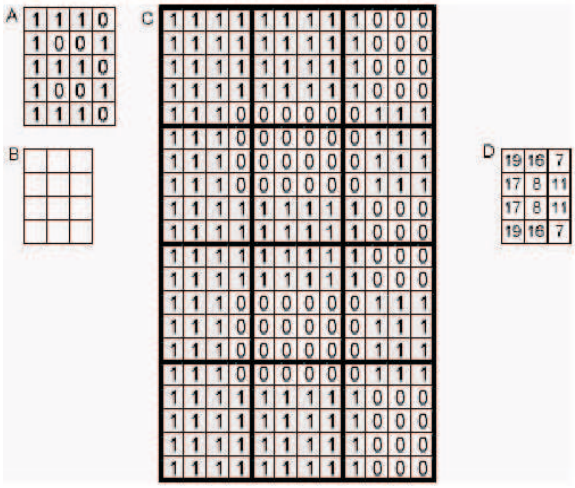
Rysunek 15. Okno dialogowe dla procedury detekcji tablicy.



Rysunek 16. Okno dialogowe dla procedury klasyfikacji znaków.



Rysunek 17. Klasyfikacja znaków: (a) metodą pomiaru odległości od linii bazowych lub (b) metodą zliczania przecięć znaku z pionowymi i poziomymi liniami.



Rysunek 18. Normalizacja rozmiaru bitmapy znaku: (a) przykład prostokątnego obszaru ze znakiem, (b) rozmiar po znormalizowaniu, (c) przekształcenie prostokąta w większy tymczasowy (jego rozmiar wyznaczony jest przez iloczyn rozmiaru aktualnego i modelowego), (d) redukcja rozmiaru tymczasowego do modelowego.

związane z ruchem ulicznym, z urządzeniami polowymi umieszczonymi na skrzyżowaniach ulic. W obecnej wersji program czujnika składa się z 4 modułów: auto-kalibracji, detekcji natężenia ruchu i długości kolejki, rozpoznawania tablic rejestracyjnych i modułu komunikacji z serwerem OMNI-MOUN.

Dalsza rozbudowa programu powinna uwzględnić zadania klasyfikacji pojazdów i ich śledzenia, celem określenia trajektorii ruchu. Możemy wyróżnić następujące klasy: samochód osobowy, autobus, ciężarówka, itp. [9]. Możliwe są tu różne rozwiązania - prosta klasyfikacja przodu pojazdu na poziomie ikoniki obrazu [9], detekcja gęstości linii i ich zorientowania [10] lub wykorzystanie 3-wymiarowego modelu [11].

W wyniku śledzenia pojazdu w krótkiej sekwencji obrazów możliwe jest określenie jego trajektorii i średniej prędkości, co pozwala na rejestrację nieoczekiwanych manewrów lub sytuacji, np. wypadku, przejechania ciągłej linii, zatrzymania na pasie ruchu, itp.

Podziękowanie

Niniejsza praca powstała w ramach projektu "Open Model for Network-Wide-Heterogeneous Intersection-Based Transport Management (OMNI)", EC-IST 1999-11250.

Literatura

[1] Grupo Etra. Informacja WWW o projekcie *Open Model for Network-Wide-Heterogeneous Intersection-Based Transport Management* (OMNI), IST 1999-11250 - na stronie koordynatora: http://www.etra.es

[2] R. Blissett. *Eyes on the road*. Roke Manor Research, IP Magazine, May/June 1992, U.K.. Strona WWW: www.roke.co.uk

[3] K. Takahashi et al.. Traffic flow measuring system by image processing. MVA'96, *IAPR Workshop on Machine Vision Applications*, Tokyo, Japan, 1996, 245-248.

[4] M. Jankowski. *Implementacja sensora ruchu drogowego*. Praca dyplomowa, IAiIS, Politechnika Warszawska, Luty 2003.

[5] W. Kasprzak, H. Niemann. Adaptive Road Recognition and Egostate Tracking in the Presence of Obstacles. *International Journal of Computer Vision*, Kluwer Academic Publ., Boston - Dordrecht - London, vol 28(1998), No. 1, 6-27.

[6] J. Balas-Cruz, J. Barroso, A. Rafael, E.L. Dagless. Real-time number plate reading. *4th IFAC Workshop on Algorithms and Architectures for Real-time Control*, Vilamoura, Portugal, April 1997.

[7] J. Barroso, A. Rafael, E.L. Dagless, J. Balas-Cruz. Number plate reading using computer vision. *IEEE International Symposium on Industrial Electronics*, Guimaraes, Portugal, July 1997.

[8] Y.G. Won, Y-K. Park. Property of greyscale hit-or-miss transform and its applications, *Machine Graphics & Vision*, vol. 9(2000), 539-547, IPI PAN, Warszawa.

[9] W. Kasprzak. An Iconic classification scheme for video-based traffic sensor tasks". W: W.Skarbek (ed.): *Computer Analysis of Images and Patterns 2001*, Springer-Vg. LNCS 2124, Berlin, 2001, 725-732.

[10] T.N. Tan, G.D. Sullivan, K.D. Baker. Fast Vehicle Localisation and Recognition Without Line Extraction and Matching. BMVC94. *Proceedings of the 5th British Machine Vision Conference*, Sheffield, U.K., BMVA Press, 1994, 85-94.

[11] W. Kasprzak. *Adaptive Erkennung von bewegten Fahrbahnobjekten in monokularen Bildfolgen mit Eigenbewegung*. Infix Publ., Sankt Augustin, Germany, 1997.