AGH Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

KATEDRA AUTOMATYKI I INŻYNIERII BIOMEDYCZNEJ

Raport na temat:

Algorytm optycznego określania poziomu cieczy

Wykonał zespół badawczy: **dr hab. inż. Paweł Rotter**
inż. Wiktor Muroń
Piotr Lizończyk

Kraków, 2015

Praca wykonana w ramach projektu CYTOROBOT finansowanego przez Narodowe Centrum Badań i Rozwoju w latach 2012-2015.
Umowa nr PBS1/A9/1/2012 w ramach Programu Badań Stosowanych w ścieżce A

Spis treści

1 Założenia

Opracowany na potrzeby projektu czujnik poziomu cieczy Move-Hough-Based Level Detector musiał spełniać następujące założenia:

- Odporność na zmiany pojemnika z cieczą.
- Odporność na zmiany koloru / naklejek na pojemniku
- Powinien dawać poziom cieczy przy lekko falującej powierzchni wody oraz w trakcie zmiany poziomu.
- Bardzo niski koszt kamerki używanej do analizy obrazu (tylko dzięki wykorzystaniu video można korzystać z niższej jakości sprzętu, gdyż przy słabym sprzęcie video, jakość jest rekompensowana przez informacje zawarte w kolejnych klatkach obrazu. Gdyby do dyspozycji był tylko aparat to niska jakość zdjęć tak czy tak uniemożliwiałaby analizę). Koszt ten musi być niski, ze względu na fakt, że potrzebnych kamerek będzie tyle ile szufladek z pojemnikami na ciecze.

W związku z powyższymi założeniami w projekcie musiano zastosować następujące rozwiązania:

- Analiza poziomu odbywa się na podstawie analizy ruchu w obrazie – pozwala to uodpornić czujnik na zanieczyszczenia stałe, zmiany kolorów, naklejek i rodzajów naczyń. Nie jest problemem, że może być używany tylko w trakcie zmiany poziomu, gdyż tylko wtedy jest potrzebny. Co do poziomu w momencie nieruchomym, to może być zapisywany pod koniec każdej operacji zmiany poziomu i pamiętany jako startowy poziom do każdej nowej zmiany poziomu.
- Podstawową technologią zastosowaną jest segmentacja pierwszego planu ruchomego z obrazu za pomocą algorytmu MOG (opisanego już w raporcie)
- Kolejnym krokiem jest analiza otrzymanego pierwszego planu, za pomocą metody transformaty Hougha, której wyniki są później odpowiednio obrabiane i interpretowane.

- Dodatkowo przetestowano pomocniczy czujnik poziomu bazujący na położeniu środka ciężkości pierwszego planu.

2 Opis algorytmu Move-Hough-Based Level Detector

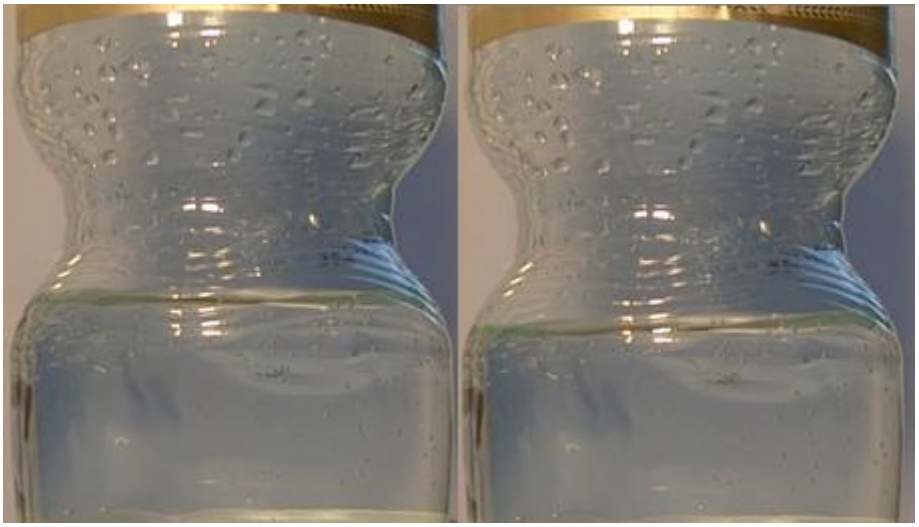
Algorytm działa w dwóch fazach głównych i fazy 0 - przygotowującej:

- Faza I – przed ustaleniem szacowanego poziomu
- Faza II – po ustaleniu szacowanego poziomu

Niezależnie od tego, w której fazie znajduje się program, wykonywane są poniższe kroki (Faza 0)

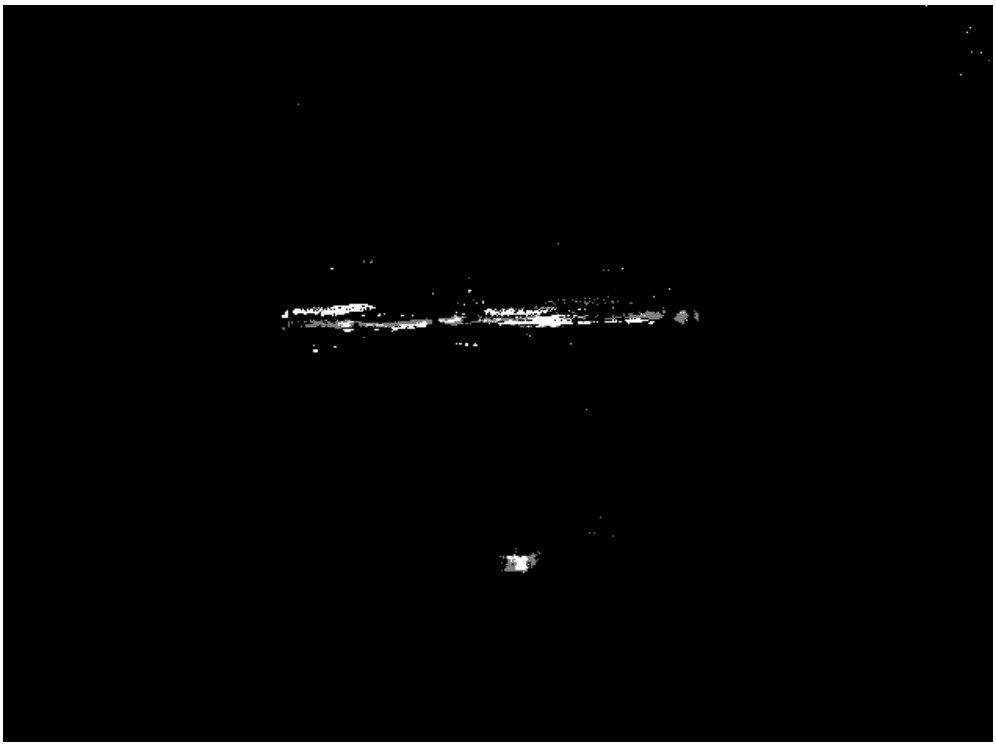
2.1 Faza 0

- Pobierz źródło video do analizy (Rys. 2-1). Do analizy zastosowano Przykładowe klatki z materiału video zawierające ruch bardzo wolno opadającego poziomu płynu w pojemniku, z powierzchnią lekko falującą ze względu na sam ruch wody.



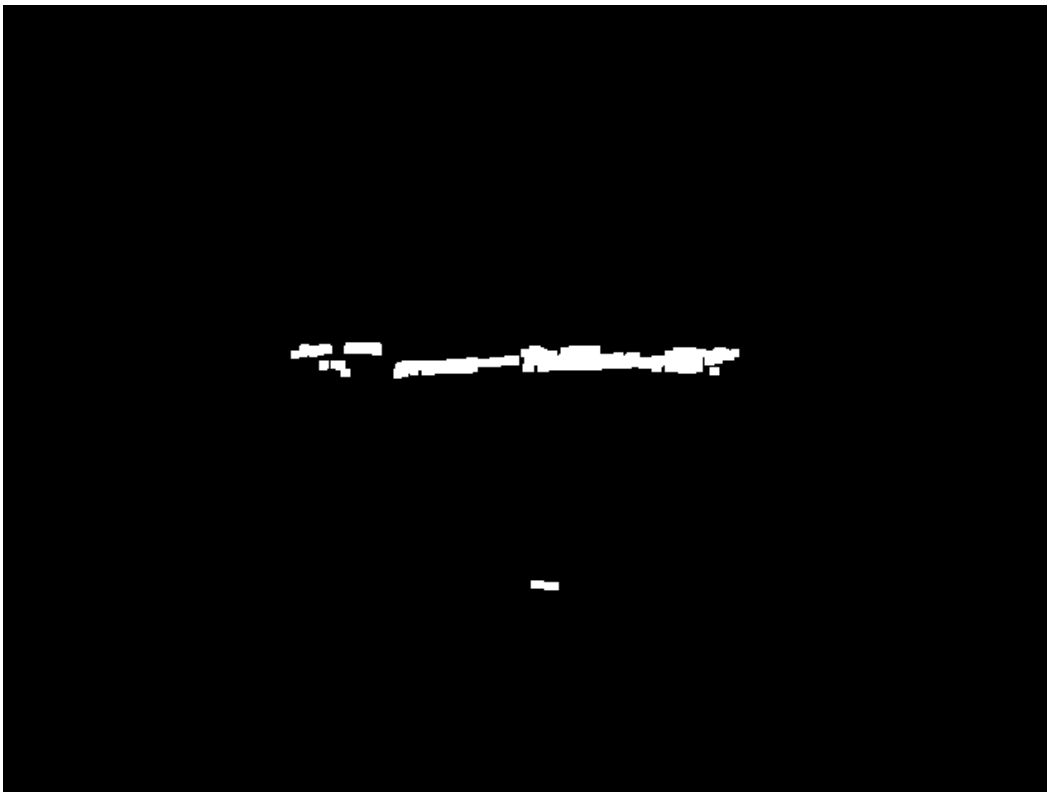
Rys. 2-1 Przykładowe klatki źródłowe

- Wydobądź z obrazu pierwszy plan (ruchomy) za pomocą algorytmu MOG2 (wielokrotnie używanego już w projekcie) – por. Rys. 2-2.



Rys. 2-2 Pierwszy plan – wynik MOG

- Zastosuj na wyniku MOG operację odszumiania za pomocą erozji i dylatacji (opisane w poprzednich częściach raportu). W końcowej fazie zastosowano dylatację o 2 razy większym kwradratowym „jądrze" niż przy erozji. Pozwoliło to na zniknięcie małych szumów i zdecydowane powiększenie fragmentów obrazu, którymi jesteśmy zainteresowani (Rys. 2-3).



Rys. 2-3 Pierwszy plan – efekt odszumienia obrazu (wynik erozji i dylatacji)

- Dla obrazu wyniku erozji i dylatacji znajdź kontury za pomocą algorytmu Canny i znajdź kontury wykryte funkcją cv::findContours
- Wykryj za pomocą transformaty Hougha na wykrytych konturach wszystkie linie (czerwone) i wybierz z nich te o niewielkich (mniejszych niż 1 stopień) odchyłkach od poziomu (zielone) (Rys. 2-4)

Rys. 2-4 Wykryte linie czerwone (wszystkie) i zielone (poziome)

- Jeżeli liczba wykrytych linii poziomych jest większa od 4 to usuwamy po 20% linii z góry i z dołu zbioru poziomych linii w celu ograniczenia błędu, a później z pozostałych wyliczana jest średnia arytmetyczna
- Wyliczona średnia linia jest wynikiem metody getLevelHough dla aktualnej klatki obrazu i nie zawsze jest faktycznym odzwierciedleniem rzeczywistego poziomu cieczy w naczyniu, np. z powodu poruszenia naczynia, szumu, czy pojawienia się bąbelków w innym miejscu

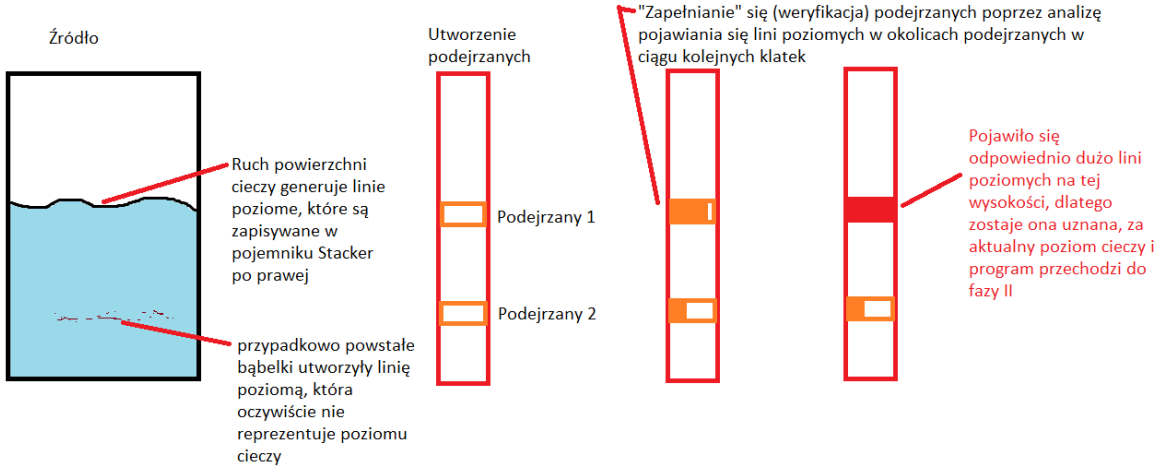
2.2 Faza I

W tej fazie jeszcze nie ustalono pierwszego szacowanego poziomu cieczy, więc w tym celu napisano klasę Stacker, która polega na przechowywaniu podejrzanych wysokości (liczb), na których pojawiają się wykryte poziome linie. W każdej kolejnej klatce:

- Jeżeli wykryty poziom po raz pierwszy lub różni się bardzo od znajdujących się do tej pory w liście podejrzanych wysokości to dodaj go do listy list w poniższy sposób
    - jeżeli różni się niewiele to dodaj go do reprezentantów danej listy tego poziomu. Poziom listy reprezentującej oblicza się jako średnią ostatnich max. 4 wpisanych do określonego poziomu liczb.
    - Jeżeli różni się wiele (parametr) to utwórz nową listę reprezentującą aktualny poziom, tworzącą nowego podejrzanego do dalszej analizy.
    - jeżeli któraś z list reprezentujących poziom przekroczy 8 wpisanych linii, to jest uznawana za poziom odniesienia i program przechodzi do Fazy II.

2.2.1 Pomocnicza klasa Stacker

Jest to lista (lub wektor) obiektów typu InertionValue, które obliczają średnią z podanych jej n (parametr inertion_number) ostatnich liczb (Rys. 2-5). Służy ona do selekcji jednego, ustalonego poziomu spośród podejrzanych miejsc, gdzie następują losowe lub nielosowe wykrycia poziomu.



Rys. 2-5 Graficznie działanie klasy Stacker:

2.3 Faza III

Mamy w tym momencie ustalony już tylko jeden wykryty poziom cieczy (zakres). Poziom ten jest przechowywany w obiekcie klasy InertionValue, który zwraca średnią z ostatnich n (parametr ustalony w programie) wartości poziomu z fazy II wysłanego do tego obiektu. Daje to swego rodzaju inercję w działaniu poprzez pamiętanie poprzednich wartości i likwiduje chaotyczne wahania. W każdej kolejnej klatce:

- Analizujemy poziom wykryty w fazie 0:
  - Jeżeli poziom wykryty w Fazie 0 nie różni się znacząco od poziomu cieczy po analizie (czyli jeden wybrany podejrzany poziom) to do obiektu InertionValue wysyłany jest aktualny poziom wykryty w Fazie 0, zastępuje on najwcześniejszą z wartości wysłanych do obiektu InertionValue, która była uwzględniana do średniej.
  - Jeżeli różni się znacząco to wyświetlany jest odpowiedni komunikat a aktualnie wykryty poziom Fazy 0 nie jest wysyłany do InertionValue wyliczającej aktualny inercyjny poziom
- Zwracamy poziom cieczy wyliczony za pomocą odpowiedniej funkcji klasy InertionValue, która daje nam uśredniony poziom z ostatnich n poziomów.

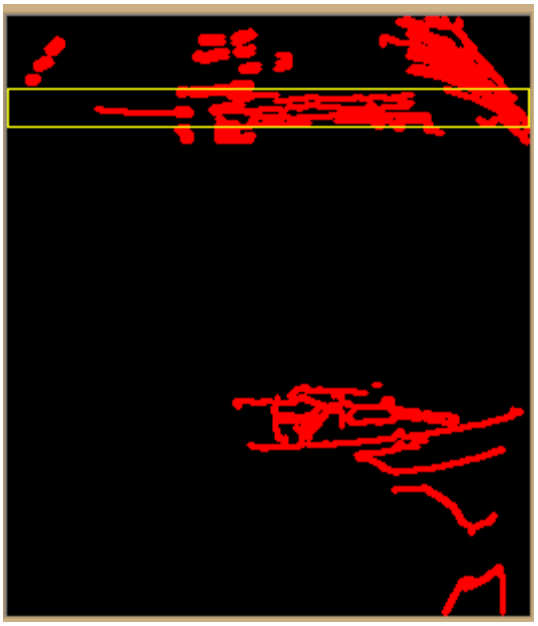
3 Pomocniczy czujnik poziomu ContourBased

W celu upewnienia się co do słuszności wykryć poziomu cieczy otrzymanego przez metodę HoughBasedLevel wymyślono dodatkową metodę weryfikującą poziom otrzymany z metody HoughBasedLevel.

Kiedy metoda HoughBasedLevel przechodzi w fazę II (czyli otrzymany jest szacowany poziom cieczy) to:

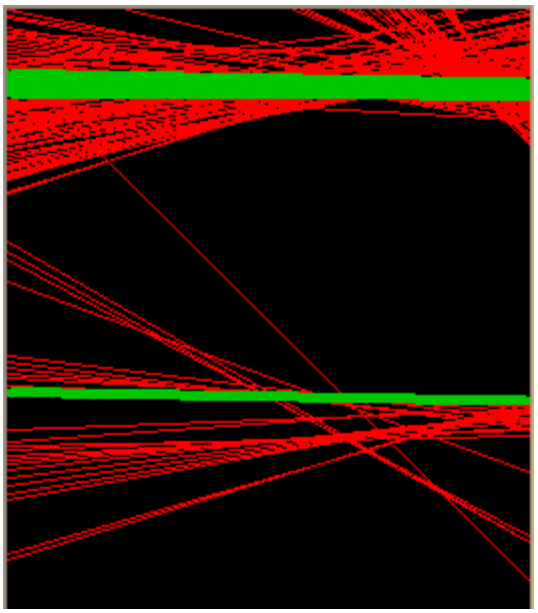
- Wyszukiwane są kontury obrazka (klatki, a nie obrazu różnicowego) otrzymane poprzez:
  - Konwersję obrazka do skali szarości za pomocą cv::cvtColor

    - Lekkie rozmycie funkcją cv::blur
    - Wykrycie konturów algorytmem Canny funkcją cv::Canny. Parametry algorytmu (np. do progowania obrazu) powinny zostać dobrane indywidualnie już podczas testów przy ostatecznym oświetleniu. W przykładowej próbce video używanej do testów parametry używane to lowThreshold = 50, highThreshold = 150, kernel_size = 3;
- Wybierany jest fragment obrazu wyszukanych konturów (nie obraz różnicowy jak w poprzedniej metodzie) wokół oszacowanego poziomu cieczy. Rozpiętość tego prostokąta zainteresowania to RANGE_WHERE_CONT = 10 w górę i w dół od szacowanego HoughBased poziomu cieczy (Rys. 3-1)



Rys. 3-1 Szukany prostokąt zainteresowania to ten w żółtym obramowaniu

- Analogicznie jak w algorytmie HoughBasedLevel detector wyszukiwane są poziome linie za pomocą transformaty Hougha na konturach klatki, które są zawarte w prostokącie zainteresowania (Rys. 3-2)



Rys. 3-2 Wykryte poziome linie

- Z wszystkich linii wybierane są wszystkie poziome, a z nich, tylko te, które znajdują się w okolicy poziomu oszacowano przez HoughBasedLevel
- Wyliczana jest średnia z wysokości otrzymanych linii poziomych. Analogicznie jak w poprzednim algorytmie wykryte w ten sposób linie są przechowywane w obiekcie InertionValue, co redukuje losowe wahania wartości poziomu wykrytego z tej metody.
- Wyniki obliczane przez ta metodę ContourBasedLevel, jeżeli wyliczone na podstawie okolic poziomu oszacowanego z obrazu różnicowego są bardzo bliskie temu poziomowi i razem skorelowane dają gotowy wynik, wraz z odpowiednią wartością błędu, który należy przyjąć ze względu na różnice wysokości cieczy wyliczone z różnych metod.

4 Rezultaty

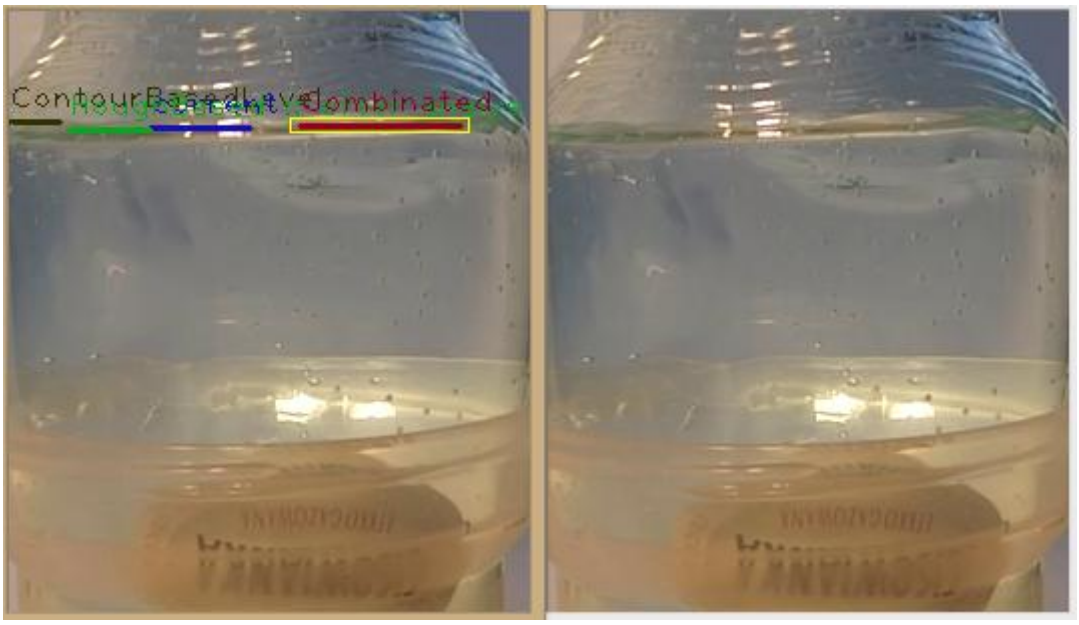
Przy wielu zróżnicowanych sposobach zmiany wysokości wody (wlewanie, wylewanie, szybkie, powolne) otrzymywano bardzo dobre i dość dokładne rezultaty. Nawet przy użyciu kamerki o bardzo słabej jakości efekty działania algorytmów nadawały się do dalszej analizy.

Próby algorytmów napisanych w ramach projektu zostały przeprowadzone na wysokiej jakości kamerze oraz na kamerce niskiej jakości. (Rys. 4-1, Rys. 4-2)

Poziomy otrzymane zostały na obrazkach ilustrujących oznaczone następująco:

- Brązowy – otrzymany z metody ContourBasedLevel z inercją
- Zielony – z metody HoughBasedLevel z inercją
- Niebieski – z metody HoughBasedLevel bez inercji (aktualny, bez pamięci poprzednich wyników)
- Czerwony – średnia ważona z poziomów ContourBased (waga 1) i HoughLevel (waga 2). Żółte obramowanie obrazuje błąd czerwonego poziomu, wysokość którego wyliczana jako 2 * różnica poziomów, z których była wyliczana.

W obu przypadkach rezultaty okazały się zadowalające.



Rys. 4-1 Próbka i efekt działania algorytmu na kamerce wysokiej jakości

Rys. 4-2 Próbka i efekt działania algorytmu na kamerce niskiej jakości