AUTOMATYKA • 2005 • Tom 9 • Zeszyt 3

Marek Zachara*

Analiza w czasie rzeczywistym ruchu kamery względem sceny na podstawie analizy wektorów ruchu

1. Wprowadzenie

Jednym z celów robotyki jest skonstruowanie autonomicznych jednostek, które zdolne będą poradzić sobie w rzeczywistym otoczeniu. W tym celu muszą one zidentyfikować obiekty znajdujące się w swoim sąsiedztwie. Dla ludzi podstawowym zmysłem wykorzystywanym do tego celu jest wzrok, ponieważ dostarcza on nam największej ilości informacji. Analogicznie, w przypadku robotów bardzo wiele istotnych informacji można uzyskać z sygnału kamery zainstalowanej na poruszającym się robocie. Analiza obrazu dostarczonego z poruszającej się kamery jest jednak zadaniem nietrywialnym. Wynika to z faktu, że w takiej sytuacji w zasadzie nie istnieją stałe punkty odniesienia, według których można porównywać kolejne klatki. W przypadku analizy obrazu ze stacjonarnej kamery opracowano wiele technik, jak choćby opisane w [7], jednak większość z nich nie może być zastosowana w omawianym przypadku.

Każdy ruch kamery względem sceny powoduje zmiany w praktycznie każdym punkcie ramki. Obrót kamery względem pionowej osi można na ogół przybliżyć liniowym przesunięciem całej ramki, natomiast ruch do przodu lub do tyłu można już tylko opisać za pomocą przekształcenia nieliniowego (elementy blisko brzegu pola widzenia będą się przemieszczać szybciej niż oddalone od niego, elementy w centrum obrazu ulegają powiększeniu lub pomniejszeniu).

Niniejszy artykuł opisuje podejście do analizy ruchu kamery względem otoczenia oparte na idei wektorów ruchu (*motion vectors*). Wektory ruchu są podstawową koncepcją wykorzystywaną przy kompresji wideo MPEG [3, 4]. Wektory te określają przemieszczenia wybranych fragmentów obrazu pomiędzy kolejnymi klatkami. Powstały nawet aplikacje pozwalające na odtworzenie ruchu kamery na podstawie danych strumienia wideo MPEG [6].

Choć w przypadku kompresji strumienia wideo nie ma znaczenia, czy ruch elementu obrazu zostanie rzeczywiście dobrze opisany (a jedynie, czy pozwoli to na zredukowanie ilości danych), to jednak idea ta po pewnych modyfikacjach dobrze nadaje się do opisu przedstawionego zagadnienia.

* Akademia Górniczo-Hutnicza, Kraków; mzachara@agh.edu.pl

W celu zapewnienia przetwarzania obrazu w czasie rzeczywistym, najistotniejszym zagadnieniem jest wykorzystanie algorytmów, które zapewnią uzyskanie odpowiednio wysokiej liczby przetwarzanych ramek na sekundę (*frame rate*).

Wbrew pozorom nie jest to tylko tautologia. Im mniejsza liczba przetwarzanych ramek na sekundę, tym większy względny ruch obiektów na scenie. Przy stałej prędkości poruszania się kamery wektor ruchu obiektu będzie dwa razy większy dla dwukrotnie mniejszej prędkości przetwarzania ramek. Wymusza to z kolei czterokrotne poszerzenie zakresu poszukiwania wektora ruchu, a zatem często czterokrotne zwiększenie ilości obliczeń.

Aby zapewnić efektywne i szybkie przetwarzanie strumienia obrazu, zaproponowano algorytm polegający na wyodrębnieniu z każdej ramki najbardziej charakterystycznych (zróżnicowanych) obszarów, których dopasowanie na kolejnej ramce będzie obarczone niską stopą błędu.

Niewielki rozmiar każdego z takich obszarów pozwala na znaczne zredukowanie obliczeń i precyzyjne ich dopasowanie. Z kolei duża liczba obszarów pozwala na stworzenie ogólnej mapy wektorów ruchu obrazu. Na podstawie tej mapy obliczany jest następnie ruch kamery względem obserwowanej sceny.

2. Wybór obszarów

Wybór obszarów do porównania na kolejnych klatkach jest kluczowy dla uzyskania precyzyjnych i wiarygodnych wyników końcowych. Obszar o niewielkim zróżnicowaniu, w szczególności np. jednolita powierzchnia, nie nadaje się do porównywania na dwóch kolejnych klatkach z tego względu, że na ogół prowadzić będzie do błędnego dopasowania. Znane są różne techniki wyboru takich obszarów, np. bazujące na cechach charakterystycznych (*features*) jak np. podane w [2], jednak ze względu na założenia szybkości przetwarzania zdecydowano się na prostą dyskretną wersję pochodnej przestrzennej jasności punktów obrazu.

Najbardziej pożądane obszary to takie, które zawierają wiele różnorodnych szczegółów, najlepiej zaś krawędzie pomiędzy obszarami o różnej jasności. W celu zlokalizowania takich obszarów, dla każdej klatki liczona jest zmiana wartości jasności pomiędzy sąsiednimi punktami obrazu w pionie i poziomie, według następującego wzoru

$$d(x,y)=\frac{1}{2}\left(\left|p(x,y)-p(x+1,y)\right|+\left|p(x,y)-p(x,y+1)\right|\right) \tag{1}$$

gdzie:

$d(x, y)$ – wartość określająca zróżnicowanie otoczenia punktu o współrzędnych x, y,
$p(x, y)$ – wartość luminancji punktu o współrzędnych x, y.

Następnie pierwotna klatka obrazu dzielona jest na pola o ustalonej wielkości i dla każdego takiego pola liczona jest uśredniona wartość zmienności przestrzennej punktów obrazu

$$k=\frac{1}{(y_t-y_b)*(x_r-x_l)}\sum_{y=y_b}^{y_t}\sum_{x=x_l}^{x_r}d(x,y) \tag{2}$$

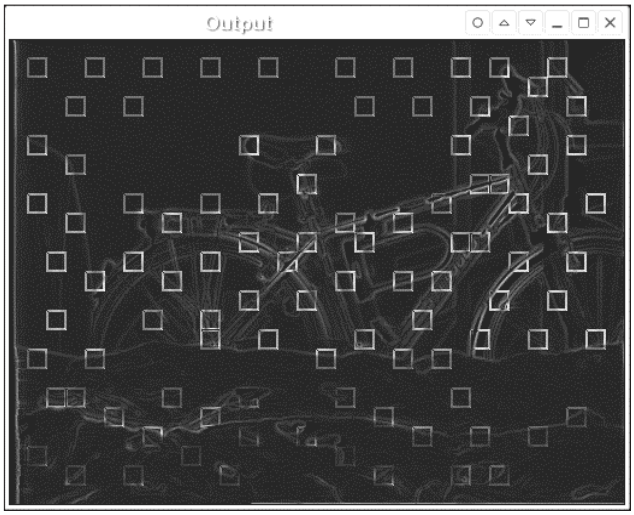
gdzie:

k – uśredniona wartość zmienności danego obszaru;
$d(x, y)$ – wartość określająca zróżnicowanie otoczenia punktu o współrzędnych x, y;
y_t, y_b, x_r, x_l – parametry określające granice obszaru (odpowiednio: górną, dolną, prawą, lewą).

W ten uzyskujemy informacje o obszarach charakteryzujących się największą zmiennością jasności, co pozwala wybrać je do dalszego przetwarzania i obliczenia mapy wektorów ruchu.

2.1. Dyskryminacja sąsiednich obszarów

Aby zapobiec sytuacji skupienia wybranych obszarów w jednym miejscu analizowanej klatki, wprowadzono dodatkową regułę do opisanego wyżej algorytmu wybierającego, polegającą na zmniejszeniu atrakcyjności obszarów znajdujących się w bezpośrednim sąsiedztwie wybranego. Dzięki temu udało się uzyskać bardziej równomierny rozkład, pozwalający na wiarygodną budowę mapy wektorów ruchu. Wynik działania tego algorytmu można zobaczyć na rysunku 1. Jak widać, algorytm efektywnie dobiera rozkład analizowanych obszarów do istniejącej sceny.



Rys. 1. Przykład działania algorytmu dokonującego wyboru obszarów

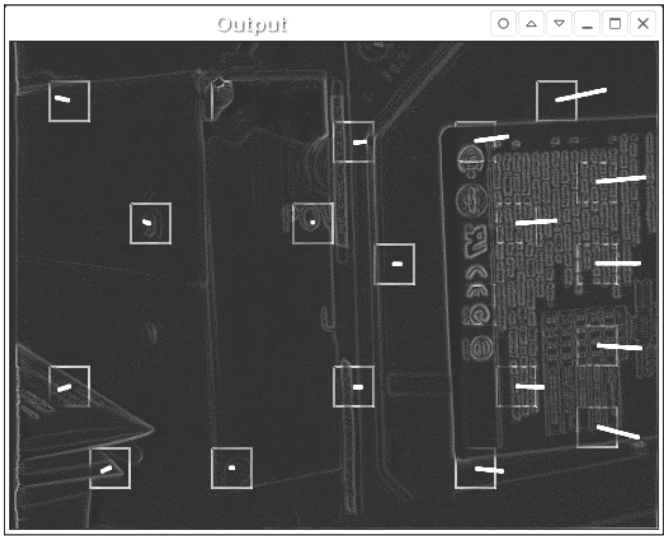
3. Dopasowanie obszarów z dwóch klatek

Wybór sposobu dopasowania obszarów na kolejnych klatkach jest kluczowy dla uzyskania odpowiedniej wydajności całego procesu. Jak wykazał profiling, procedura ta zajmuje od 50÷90% całego czasu przetwarzania. Poszerzenie zakresu przeszukiwania, jak

było to już wspomniane wcześniej, zwiększa wykładniczo liczbę operacji, z drugiej strony pozwala jednak prawidłowo rozpoznać znacznie szybsze ruchy sceny – a zarazem zmniejszyć liczbę błędów.

W celu ograniczenia liczby operacji przy jednoczesnym zachowaniu dużego pola przeszukiwania, zastosowano metodę wielokrotnych przybliżeń (tzw. piramidkę) opisaną m.in. w [1] i [5]. Polega to na dopasowaniu regionu w kilku skalach (poczynając od najmniejszej). W najmniejszej skali (przykładowo pomniejszenie 16×) przeszukiwany i dopasowywany jest cały obszar, natomiast potem kolejno w wyższych skalach przeszukiwanie odbywa się jedynie w otoczeniu wektora ruchu znalezionego w skali niższej. Dzięki temu w wyższych skalach dokonywane jest jedynie 9 przeszukań najbliższego sąsiedztwa. Zmniejsza to istotnie ilość niezbędnych operacji i pozwala na zwiększenie efektywności całego procesu.

Końcowym efektem tego etapu przetwarzania jest uzyskanie listy składającej się z wektorów ruchu dla każdego z wybranych obszarów. Wizualizacja tych wektorów została przedstawiona na rysunku 2. Taki sposób analizy obrazu jest określany w literaturze jako *optical flow*.



Rys. 2. Wizualizacja wektorów ruchu dla przypadku ruchu kamery na wprost z równoczesnym lekkim obrotem w lewo

4. Korekcja dopasowania – „mapa ciągłości”

W przypadku analizy statycznej sceny, można zastosować dodatkowy mechanizm, który pozwoli na zwiększenie dokładności i wiarygodności procesu odtworzenia ruchu kamery.

Mechanizm ten, nazwany przez autora „mapą ciągłości” opiera się na założeniu, że w przypadku statycznej sceny przemieszczenie każdego obszaru pomiędzy kolejnymi

klatkami jest zbliżone do przemieszczenia obszarów sąsiadujących. Oczywiście dopuszczalne są różnice, jednak wektory ruchu sąsiednich obszarów są podobne co do wartości i kierunku.

Wychodząc z tego założenia, w kolejnym kroku przetwarzania dla każdego obszaru, dla którego obliczony został wektor ruchu, wybierane zostają najbliżej położone sąsiednie obszary, a następnie porównywane są ich wektory ruchu. Ostateczny wektor ruchu dla danego obszaru uwzględnia z odpowiednimi wagami pierwotnie wyliczony wektor ruchu, jak również wektory ruchu wybranych sąsiadów

$$\vec{c} = \frac{1}{w_o + r * w_n} \left(w_o * \vec{v}_o + \sum_{i=1}^{r} w_n * \vec{v}_i \right) \tag{3}$$

gdzie:

c – wynikowy wektor ruchu badanego obszaru,
v_o – pierwotnie znaleziony wektor ruchu badanego obszaru,
r – liczba branych pod uwagę sąsiednich obszarów,
v_i – wektor ruchu danego sąsiedniego i-tego obszaru,
w_0 – waga wektora ruchu danego obszaru,
w_n – waga sąsiednich wektorów ruchu.

Jak wykazały doświadczenia, wykorzystanie tej metody pozwala skutecznie dyskryminować wektory ruchu będące wynikiem błędnego dopasowania obszaru na sąsiednich klatkach – w efekcie zwiększając spójność całej mapy wektorów ruchu. Dobre efekty doświadczalne uzyskano dla czterech sąsiednich obszarów przy $w_s = 3$ i $w_s = 1$.

5. Obliczanie ruchu kamery

Uzyskana mapa wektorów jest ostatecznie analizowana w celu określenia ruchu kamery.

Ocena obejmuje trzy podstawowe możliwe przemieszczenia:

1) przód/tył,
2) obrót lewo/prawo,
3) obrót góra/dół.

Przy zastosowaniu opisanej techniki możliwe jest też łatwe określenie rotacji kamery względem osi optycznej, choć dotychczas nie było to wykorzystywane.

Ocena ruchu odbywa się poprzez dopasowanie wektorów ruchu do odpowiednich wzorców. Dla przykładu: wektory ruchu skierowane w lewo w lewej części klatki i w prawo w prawej części klatki, ze wzrostem wartości wektorów w miarę oddalania od centrum oznaczają ruch kamery w przód. Współrzędna pozioma uśrednionego wektora wszystkich wektorów ruchu określa obrót kamery lewo/prawo itp.

W celu obliczenia bezwzględnych wartości ruchu kamery (tj. w centymetrach czy stopniach) niezbędne jest przeprowadzenie doświadczeń i kalibracji współczynników dla konkretnego zastosowanego sprzętu.

Wpływ bowiem na to ma wiele czynników, m.in.:

– rozdzielczość,
– wielkość matrycy,
– długość ogniskowej.

6. Podsumowanie

W niniejszej pracy przedstawiony został sposób dynamicznego doboru obszarów porównywania kolejnych ramek wideo, pozwalający na osiągnięcie dużej prędkości przetwarzania przy zachowaniu wysokiej precyzji analizy. Przedstawiony proces oceny ruchu kamery pozwalał w doświadczeniach praktycznych na osiągnięcie stabilnych i wiarygodnych wyników. Szczególnie zastosowanie dużej ilości małych obszarów dopasowywanych niezależnie (jak na rys. 1) pozwoliło podnieść jakość procesu ze względu na zminimalizowanie wpływu pojedynczych błędnych dopasowań.

Wybór małych obszarów pozwala na uwzględnienie rotacji kamery według dowolnej osi – ponieważ przy dużej szybkości przetwarzania, a co za tym idzie – małych zmianach położenia elementów sceny oraz przy niewielkim rozmiarze dopasowywanych elementów, każdy z tych ruchów może być z dobrym przybliżeniem potraktowany jako przemieszczenie liniowe. Pozwala to na uniknięcie czasochłonnych obliczeń związanych z przekształceniem nieliniowym.

Aktualnie dalsze prace przy wykorzystaniu opisanego algorytmu są skoncentrowane na odtworzeniu głębi przestrzeni (3D) za pomocą analizy wektorów ruchu poszczególnych elementów obrazu względem kamery.

Literatura

[1] Burt P.J. *et al.*: *The Laplacian pyramid as a compact image code*. IEEE Trans. Commun., vol. COM-31, 1983, 532–540
[2] Espiau F. *et al.*: *Robust features tracking for robotic applications*. http://www-sop.inria.fr/icare/personnel/malis/papers/malis_ICRA-02_tr.pdf
[3] ISO-11712 *Coding of Moving Pictures and Associated Audio (MPEG1)*
[4] ISO-13818 *Generic Coding of Moving Pictures and Associated Audio (MPEG2)*
[5] Minh N. Do. *et al.*: *Framing Pyramids*. IEEE Trans. On Signal Processing, vol. 51, No. 9, 2003
[6] Pilu M.: *On Using Raw MPEG Motion Vectors To Determine Global Camera Motion*. HP Laboratories Bristol, 1997, http://www.hpl.hp.com/techreports/97/HPL-97-102.pdf
[7] Viet H. *et al.*: *Recognition of Motion in Depth by a Fixed Camera*. Proc. 7th Digital Image Computing: Techniques and Applications, Sydney, 2003