FPGA TABANLI PARALELLEŞTİRİLMİŞ PLESSEY KÖŞE TESPİTİ

M. Fatih AYDOĞDU [(a)], M. Fatih DEMİRCİ [(b)], Coşku KASNAKOĞLU [(c)]

[(a)] TOBB ETÜ, E. Elektronik Müh. Böl. 06560, Ankara, mfatihaydogdu@gmail.com

[(b)] Yrd. Doç. Dr. TOBB ETÜ, Bilgisayar Müh. Böl. 06560, Ankara, mfdemirci@etu.edu.tr

[(c)] Yrd. Doç. Dr. TOBB ETÜ, E. Elektronik Müh. Böl. 06560, Ankara, kasnakoglu@etu.edu.tr

ÖZET

Bu makalede, Harris köşe tespiti algoritmasının sahada programlanabilir kapı dizileri(FPGA) üzerinde paralelleştirilerek gerçek zamanlı uygulamalara uygun hale getirilmesi anlatılmaktadır. Çalışmanın amacı, mobil bir robot üzerinde gerçekleştirilmesi planlanan eş zamanlı konumlama ve haritalama(EZKH) projesinde kullanılmak üzere güvenilir bir köşe tespiti algoritmasını Xilinx'e ait Virtex-5 XC5VLX50FFG676 FPGA üzerinde uygulayıp gerçek zamanlı dijital bir yapı oluşturmaktır. Çalışma robotun da kontrolünde kullanılacak olan Xilinx'e ait ML501 kartı üzerinde gerçekleştirilmiştir. Görüntüleri elde etmek için OmniVision'ın OV7720 renkli görüntü sensörü kullanılırken, ham ve işlenmiş görüntüleri depolamak için ML501 kartı üzerinde bulunan MICRON'a ait MT4HTF3264HY-53E DDR2 SODIMM bellek kullanılmıştır.

Anahtar Kelimeler: FPGA, Köşe Tespiti, Renkli Görüntü İşleme

ABSTRACT

In this article, the appropriation for the real time implementations of Harris corner detection algorithm by parallelizing it on field programmable gate array(FPGA) is presented. The goal of the study is to costruct a real time digital structure implementing a reliable corner detection algorithm on Xilinx's Virtex-5 XC5VLX50FFG676 FPGA to be used in the simultanous localization and mapping project planned to be realized on a mobile robot. The study is carried out on ML501 card which will be also used in the control of the robot. While OV7720 sensor, OmniVision's color image sensor is used to acquire the images, to store raw and processed data the memory, MT4HTF3264HY-53E DDR2 SODIMM from MICRON, which is also on the ML501 card is used.

Keywords: FPGA, Corner Detection, Color Image Processing

1.GİRİŞ

Dijital görme günümüzde başta askeri, sağlık ve uzay araştırmaları olmak üzere birçok stratejik alanda yaygın olarak kullanılmaktadır. Görüntü sensörleri sağladıkları bilginin içeriğinin zengin olması sebeyiyle dijital görme

sistemlerinde tercih edilmektedir. Sağlanan bu zengin içerik artan işlem gücü gereksinimlerini de beraberinde getirmektedir. Bu durum özellikle sınırlı hacimleri ile mobil robotlar için sorun oluşturmaktadır. Robotların işlem gücü FPGA'larla veya çok çekirdekli işlemcilerle arttırılabilmektedir.

Dijital görme sistemlerinde çoğu kez belirli bir paterne ait bilgi aranırken, köşe noktaları bu paternerlerin yapı olarak en basitleri arasındadır. Köşe tespiti algoritmaları kenar tespiti tabanlı olanlar ve doğrudan köşe tespiti yapanlar olarak ikiye ayrılabilir. Mokhtarian'ın[1] eğim ölçekli uzayını(CSS) kullandığı köşe tespit yöntemi ilk türün başarılı bir örneğidir. İkinci türün en basit örneklerinden olan Moravec[2] algoritması Harris[3] tarafından geliştirilerek Plessey algoritması oluşturuldu. CSS algoritması ve onun geliştirilmiş versiyonu doğrudan köşe tespiti yapan algoritmalara göre daha iyi sonuç verse de[4] algoritmanın gerçek zamanlı uygulamalara uygun hale getirilmesi diğer algoritmalara göre daha zordur. Yaygın olarak kullanılan ve doğrudan köşe tespiti yapan algoritmalar arasında Harris algoritması diğerlerinden daha iyi sonuçlar vermektedir[5][6][7]. Siyah beyaz görüntüler için geliştirilen Plessey algoritmasını Montesinos[8] renkli görüntülerde kullandı.

Plessey algoritmasını gerçek zamanda çalışan sitemlere uygun hale getirmek için FPGA'larda bulunan paralelleştirilebilir yapı uygundur. Houben[9] trafik gözetimi amaçlı oluşturduğu sisteminde Plessey algoritmasını FPGA üzerinde uygulayarak köşe tespiti gerçekleştirdi. Çalışmada 752'ye 480 çözünürlükteki siyah beyaz görüntüden köşelerin tespit edilmesi 9ms sürmektedir.

Bu makalede, Plessey algoritmasını renkli görüntülerde çalışacak şekilde modifiye edilmiş halinin Xilinx firmasına ait Virtex-5 XC5VLX50FFG676 FPGA çekirdeği üzerinde paralelleştirilerek gerçek zamanlı uygulamalara uygun hale getirilmesi tartışılmıştır. Paralelleştirilen yapı yürütülmekte olan mobil bir robot üzerinde EZKH projesinde kullanılacaktır. Robotta bulunacak stereo kameralardan gelen görüntüler üzerinde köşe tespiti yapıldıktan sonra belirlenen köşeler arasında eşleme yapılarak köşelerin robota göre 3 boyuttaki konumu tespit edilecektir. Tespit edilen konum bilgisi aynı FPGA üzerinde oluşturulacak EZKH modülüne aktarılarak modüle konum bilgisi sağlanacaktır.

2. RENKLİ GÖRÜNTÜLER İÇİN PLESSEY(HARRİS) ALGORİTMASI

Plessey algoritması Moravec algoritmasındaki eksikliklerin giderilmesi sonucu oluşturulmuştur. Moravec algoritmasında temel olarak görüntünün belirli bir alanının parlaklığının çevresine göre ne kadar farklı olduğunu hesaplanarak köşe olan pikselleri diğer piksellerden ayırır. Görüntüdeki bir pikselin köşe olup olmadığına karar vermek için o pikseli merkez kabul eden ve piksele belirli bir yakınlıkta kalan kare alandaki komşularının parlaklığı referans kabul edilir. Bu alanın aralarında 45° fark olan 4 farklı yöne kaydırılması sonucu kapsanan piksellerin parlaklıkları ile alandaki piksellerin parlaklıkları arasındaki fark hesaplanır.

$$F_{u,v}(x,y) = \sum_{m,n} w(m,n)(I(x+m,y+n) - I(x+m+u,y+n+v))^2 \quad (1)$$

Denklemde $F_{u,v}(x,y)$ (x,y) merkezli alanın piksellerinin parlaklık değerleri ile, alanın (u,v) yönünde kaydırılması sonucu ulaşılan alandaki piksellerin parlaklıkları arasındaki farkı göstermektedir. Denklemde m ve n ise alanın yatay ve dikeydeki büyüklüğüdür. Alandaki piksellerin çarpılacağı katsayıları belirleyen w(m,n) fonksiyonu alan içinde bir birim iken alan dışında sıfırdır. Her bir piksel için hesaplanan 4 farklılık değerinden en küçüğü o piksel için köşe olma ölçüsü(KOÖ) olarak kabul edilir. KOÖ'leri deneysel olarak belirlenen bir eşik değerinin üzerindeki pikseller Moravec algoritmasına göre köşe özelliği göstermektedirler.

Harris[3] 1988'de Moravec algoritmasındaki 3 eksikliği dikkate alarak köşe ve kenar tespiti için yeni bir algoritma geliştirdi. Harris, Moravec algoritmasının sadece 45°'lik kaymalardaki değişimleri dikkate aldığı için köşe tespitinin yönden bağımsız sonuçlar vermedigini bunun için tüm yönlerdeki değişimerle bakılması gerektiğini belirtti. Belirli bir (u,v) yönü herhangi bir (Δu, Δv) yönü olacak şeklinde yazılır.

$$F_{\Delta u,\Delta v}(x,y) = \sum_{m,n} w(m,n)(I(x+m,y+n) - I(x+m+\Delta u,y+n+\Delta v))^2 \quad (2)$$

Eğer I_x ve I_y kısmi türevleri ifade ediyorsa, taylor açılımı kullanarak

$$I(x+m+\Delta u,y+n+\Delta v) \approx I(x+m,y+n) + I_x(x+m,y+n)\Delta x + I_y(x+m,y+n)\Delta y \quad (3)$$

2. eşitlik

$$F_{\Delta u,\Delta v}(x,y) = \sum_{m,n} w(m,n)(I_x(x+m,y+n)\Delta x + I_y(x+m,y+n)\Delta y)^2 \quad (4)$$

haline dönüşür. Harris'in Moravec algoritmasında gösterdiği ikinci eksiklik ise algoritmada kullanılan w(m,n) fonksiyonunun yanlızca iki değer alması sebebiyle algoritmanın gürültüye karşı dayanıksız olmasıdır. Alan merkezine uzakta bulunan piksellerin bu sorun sebebiyle oluşturacakları gürültüleri w(m,n) fonksiyonunu gaussian dağılımında seçerek engellemiştir.

$$w(m,n) = \exp\frac{-(m^2+n^2)}{2\sigma^2} \quad (5)$$

$$F_{\Delta u,\Delta v}(x,y) = \sum_{m,n} \exp\frac{-(m^2+n^2)}{2\sigma^2}(I_x(x+m,y+n)\Delta x + I_y(x+m,y+n)\Delta y)^2 \quad (6)$$

Eşitlik sadeleştirilirse 6. Eşitlik

$$F_{\Delta u,\Delta v}(x,y) = \sum_{W} (I_x\Delta x + I_y\Delta y)^2 \quad (7)$$

Halini alır. 7. eşitlik matris yapısına dönüştürülebilir.

$$F_{\Delta u,\Delta v}(x,y) = \begin{bmatrix}\Delta x & \Delta y\end{bmatrix} M(x,y) \begin{bmatrix}\Delta x \\ \Delta y\end{bmatrix} \quad (8)$$

$$M(x,y)=\begin{bmatrix}\sum_W I_x{}^2 & \sum_W I_x I_y\\ \sum_W I_x I_y & \sum_W I_y{}^2\end{bmatrix}=\begin{bmatrix}A & C\\ C & B\end{bmatrix} \quad (9)$$

Harris, diagonal olan M matrisinin özdeğerlerinin yerel otokorelasyon fonksiyonunun kavislenmesi ile orantılı olduğunu matrisin özdeğerlerinden ikisinin de büyük olmasının o pikselin köşe olduğuna, sadece birinin büyük olmasının ise pikselin kenar olduğuna işaret ettiğini gösterdi. Böylece Moravec algoritmasındaki 3.sorun olan kenar ve köşe noktaların yakın KOÖ'lerine sahip olması sorununu çözüldü. Harris, M matrisinin özdeğerlerinin hesaplanmasının fazla zaman alacağından daha kolay hesaplanabilen bir köşe veya kenar olma ölçüsü(KKOÖ) kullandı. Burda k değiştirilebilinen bir sabittir.

$$\text{KKOÖ}=\det(M)-k.Tr(M)^2=AB-C^2-k(A+B)^2 \quad (10)$$

Burada elde edilen KKOÖ pozitif ve belirli bir büyüklükte ise (x,y) pikseli köşeye, negatif ve belirli bir büyüklükte ise kenara karşılık gelmektedir.

Montesinos ise Harris'in geliştirdiği yöntemi renkli görüntülerde kullandı. Renkli görüntülerde M matrisi şu şekilde oluşturulur.

$$M(x,y)=\begin{bmatrix}\sum_W K_x{}^2+Y_x{}^2+M_x{}^2 & \sum_W K_xK_y+Y_xY_y+M_xM_y\\ \sum_W K_xK_y+Y_xY_y+M_xM_y & \sum_W K_y{}^2+Y_y{}^2+M_y{}^2\end{bmatrix}=\begin{bmatrix}A & C\\ C & B\end{bmatrix} \quad (11)$$

Burada K_x, Y_x, M_x sırasıyla piksellerin kırmızı, yeşil ve mavi parlaklıklarının x yönündeki değişimleri iken K_y, Y_y, M_y ise piksellerin y yönündeki değişimleridir.

Tablo-1. Gradyan çarpımlarını bulanması aşaması

```
1:      Gradyan_Çarpımlarını_Bul( Ik, Iy, Im, M)
2:          m ← sizeof(M)+1;
3:          for i ← kırmızı, yeşil ve mavi parlaklık değerleri
4:              for a ← 1 to sizeof(M)
5:                  for b ← ((m/2)+1) to  sizeof(M)
6:                      Gi,x ← ( M(a,b)*(Ii(a,b)- Ii(a,m-b) ) + Gi,x;
7:                      Gi,y ← ( M(b,a)*(Ii(b,a)- Ii(m-b,a) ) + Gi,y;
8:                  endfor
9:              endfor
10:             Gi,x ← Gi,x/sum(M);
11:             Gi,y ← Gi,y/sum(M);
12:         endfor
13:         Gçx ← Gk,x* Gk,x + Gy,x* Gy,x + Gm,x* Gm,x;
14:         Gçy ← Gk,y* Gk,y + Gy,y* Gy,y + Gm,y* Gm,y;
15:         Gçxy ← Gk,x* Gk,y + Gy,x* Gy,y + Gm,x* Gm,y;
16:         return (Gçx, Gçy, Gçxy);
17:     endfunction
```

Köşe tespitinin simlasyonu ve donanım üzerinde gerçeklenmesi için algoritma iki aşamaya ayrılmıştır ve bu aşamalar Tablo-1 ve Toblo-2'de gösterilmiştir.

Tablo-1'de gösterilen ilk aşamaya gradyan maskesinin katsayılarını belirten M matrisi ile birlikte bu matrisin boyutlarındaki bir alanda bulunan piksellerin kırmızı(I_x), yeşil(I_y) ve mavi(I_m) renklerdeki parlaklık değerlerini içeren matrisler girdi olarak girer. Bu alanın merkezindeki pikselin gradyan çarpımları ($Gç_x$, $Gç_y$ ve $Gç_{xy}$) bu aşamadan çıktı olarak çıkmaktadır.

Tablo-2. KKOÖ'lerinin bulunması aşaması

```
1:      Kenar_Köşe_Olma_Ölçüsünü_Çıkar(GÇx, GÇy, GÇxy, N, k)
2:          for i ← 1 to sizeof(N)
3:              for j ← to sizeof(N)
4:                  A ← (N(i,j)*GÇx(i,j))+A;
5:                  B ← (N(i,j)*GÇy(i,j))+B;
6:                  C ← (N(i,j)*GÇxy(i,j))+C;
7:              endfor
8:          endfor
9:          A ← A/sum(N);
10:         B ← B/sum(N);
11:         C ← C/sum(N);
12:         KKOÖ ← A*B – C^2-k*(A+B)^2;
13:         return KKOÖ;
14:     endfunction
```

Tablo-2'de ifade edilen ikinci aşamada ise 11. eşitlikte belirtilen W penceresinin katsayılarını gösteren N matrisi, bu matrisin boyutlarındaki bir alan için ilk aşamada hesaplanmış olan gradyan çarpımları içeren $GÇ_x$, $GÇ_y$, $GÇ_{xy}$ matrisleri ve 10. eşitlikteki k sabiti girdi olarak girer. Bu aşama sonunda alanın merkezindeki pikselin KKOÖ hesaplanmış olur.

3. ALGORİTMANIN PARALELLEŞTİRİLMESİ

3.1. TEST AŞAMASI

Algoritmayı paralelleştirme aşamasına geçmeden önce görüntü sensörü ile ML501 kartı arasındaki bağlantılar sağlanarak sensörün RGB555 formatında veri aktarması sağlanmıştır. Gelen veri kartta bulunan DVI sürücü entegresi ile monitöre aktarılmıştır. Daha sonra kameradan alınan anlık görüntüler kartta bulunan DDR2 belleğe kaydedilmiştir. Kaydedilen bilgiler Xbee kablosuz haberleşme modülleriyle bilgisayar ortamına aktarılmış ve Matlab programı ile test görüntüleri bilgisayarın belleğine kaydedilmiştir.

Bilgisayar belleğine kaydedilen test görüntüleri yardımıyla algoritmada kullanılacak olan sabitlerin değerleri benzetim ortamında belirlenmiştir.

Test görüntüsü üzerinde yapılan çalışmalarda gradyan hesaplamasında ve Harris algoritmasındaki pencere katsayıları için değişkesi 1,2 olan bir gaussian

dağılımın en iyi sonuç verdiği gözlemlendi. Gradyan hesabında guassian dağılımına sahip bir maskenin kullanılması gürültüyü azaltma amaçlıdır. Yine bu iki pencerenin büyüklüğünün 5x5 olması durumunda RGB555 formatındaki görüntüler için şekil-1'de görüldüğü gibi yeterince yönden bağımsız köşe tespitinin yapılabildiği gözlemlendi. 5x5'lik bir pencere büyüklüğü ve 1,2'lik değişkeye sahip ayrık 2 boyutlu gaussian dağılımının katsayılarının tam sayıya dönüştürülmüş hali şekil-1'de gösterilmiştir. Harris algoritmasındaki k sabiti hem donanımsal olarak uygulamasının kolaylığı açısından hem de algoritmadaki başarısı sebebiyle 1/16 olarak belirlendi.

1	2	3	2	1
2	6	8	6	2
3	8	12	8	3
2	6	8	6	2
1	2	3	2	1

Şekil-1. Test görüntüsü ve katsayıları tamsayı olan gaussian penceresi

3.2. TASARIM

Donanım tümüyle tam sayılar üzerinde çalışacak şekilde oluşturulmuştur. Tablo-1 ve tablo-2'de belirtilen algoritma donanıma aktarılırken de 2 ayrı modül şeklinde tasarlanmıştır. Bunun temel sebebi 2. aşamada bir pikselin KKOÖ'nü hesaplamak için o pikselin ve M boyutlarındaki alanda kalan komşularının gradyan çarpımlarına ($Gç_x$, $Gç_y$ ve $Gç_{xy}$) ihtiyaç duyulmasıdır. Dolayısıyla oluşturulan 2 modülden birincisi piksellerin gradyan çarpımlarını hesaplarken ikinci modül de bu gradyan çarpımlarıyla KKOÖ'lerini hesaplamaktadır.

Donanım tasarlanırken toplama ve çıkarma işlemleri için Xilinx'in ISE webpack programının donanım sentezlemesine izin verilirken çarpma işlemleri için yine Xilinx'in core generator programıyla modüller oluşturulmuştur. Hem fazla sayıda saat vuruşu(SV) gerektirmesi hem de donanımsal olarak fazla yer kaplaması sebebiyle bölme modülleri kullanmaktan kaçınılmıştır. Bunun yerine SV gerektirmeyen kayma işlemleriyle işlenen verinin büyüklüğü azaltılmıştır. Bu durum algoritmanın çalışmasında herhangi bir soruna sebep olmazken sadece KKOÖ'lerinin kıyaslanacağı eşik değerini değiştirmektedir.

İlk aşamanın donanımında, ilk SV'da tablo-1'in 6. ve 7. satırlarındaki çıkarma işlemleri paralel olarak 3 renk için ve her iki yönde de gerçekleştirilir. Bu satırlardaki çarpma işlemlerinden 1, 2 ve 8 katsayıları ile yapılanlar kayma işlemleri ile SV gerektirmeden yapılırken 3 ve 6 katsayıları ile yapılacak çarpmalar için bir SV gecikmeli modüller oluşturulmuştur. Sentezlenen toplama modülleriyle 6. SV'da tüm gradyanlar hesaplanır. 10. ve 11. satırlardaki bölme işlemleri gecikmeyi engellemek için kayma işlemleri ile SV gerektirmeden yapılır. 13 ile 15 satırlar arasındaki gradyan çarpımları paralel

olarak 3 SV gecikmeye sahip çarpıcılarla gerçekleştirilir ve sonuçlar toplama modülleriyle 12 SV'da elde edilir.

İkinci aşamanın donanımında, ilk SV'dan 6. SV'na kadar ikinci tablonun 2 ile 8. satırları arasında hesaplanan A, B ve C değerleri hesaplanır. 9. ve 11. Satırlar arasındaki bölme yine kayma şeklinde yapılarak SV gecikmeleri engellenir. 7.SV'da 12. satırdaki AXB ve CXC çarpma işlemlerine 3 SV gecikmeye sahip 2 DSP modülünde başlanırken A+B toplama işlemi de bu SV'da yapılır. 8. SV'da başka bir DSP modülüyle A+B toplamasının karesi alınmaya başlanır. Benzetim aşamasında k sabiti 1/16 olarak seçildiği için k ile çarpma işlemi SV gerektirmeden kayma işlemiyle gerçekleştirilir. 12. satırdaki çıkarma işlemleri 12. ve 13. SV'da gerçekleştirilir ve 2. aşama 13 SV'da tamamlanmış olur.

Elde edilen KKOÖ eşik değerinin altında ise belleğe o pikselin KKOÖ 0 olarak yazılır. Bellekteki KKOÖ'lerine 8x8'lik bir alanda maksimum olmayanı baskılıma yöntemi uygulanır. Bu alandaki KKOÖ 0'dan farklı olan piksellerden en büyük KKOÖ'ne sahip olan piksel köşe olarak saptanır.

4. PERFORMANS

Sentez sonucu 12 SV'da tamamlanan 1. aşamanın 463MHz'lik maksimum çalışma frekansına sahip olacağı tespit edilirken, 13 SV'luk 2. aşamanın 373MHz'lik bir maksimum çalışma frelkansına sahip olacağı tespit edilmiştir. Buna göre bu modüllerden sadece 1er tane kullanarak bile 640'a 480 çözünürlükteki, RGB555 formatındaki görüntünün köşelerini tespiti 17,3ms sürmektedir. Bu modüllerden bir kaç tanesi parallel olarak kullanılarak çalışma süresi azaltılabilir.

Algoritmanın paralelleştirilmiş bu hali, tablo-3'de görüldüğü gibi XC5VLX50 gibi küçük bir çekirdeğinin %20'sinden az bir kısmını işgal etmektedir. Bu çekirdek çalışmanın geri kalanını da düşünerek daha fazla paralelleştirme imkanı vermemektedir. Ancak oluşturulan modüller daha büyük FPGA'lara paralel bir şekilde gömülerek köşe tespit süresi çok daha düşük seviyelere çekilebilir.

Tablo-3. Oluşturulan modüllerin kullandığı kaynak miktarları

	1.Aşama		2.Aşama		Toplam		XC5VLX50
	Adet	%	Adet	%	Adet	%	Adet
Dilim Belleği	1834	6,37	3186	11,06	5020	17,43	28800
LUT	1912	6,64	2377	8,25	4289	14,89	28800
DSP48	0	0	3	6,25	6,25	6,25	48

5. SONUÇ

Nesne tespiti, hedef takibi ve konum belirleme gibi görüntü tabanlı çalışan sistemlerin gerçek zamanlı çalışması hayati önem arz etmektedir. FPGA'lar paralel olarak kullanılabilinen işlem güçleriyle bu sistemlerin gerçek zamanlı çalışmalarını sağlamaktadırlar. Bu çalışmada görüntü tabanlı uygulamalarda sıkça kullanılan Plessey köşe tespiti algoritması renkli görüntüler için Xilinx firmasına ait XC5VLX50 FPGA'sı üzerinde paralelleştirilerek gerçek zamanlı çalışmaya uygun hale getirilmiştir. Oluşturulan yapı EZKH yapacak olan, stereo kameraya sahip mobil bir robot üzerinde kullanılacaktır.

6. REFERANSLAR

[1] F. Mokhtarian, R. Suomela, "Robust Image Corner Detection Through Curvature Scale Space," IEEE Transactions on Pattern Analysis and Machine Intelligence, 20, 12, (1998).

[2] H. Moravec, "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover," tech. report CMU-RI-TR-80-03, Robotics Institute, Carnegie Mellon University, Stanford University, (1980).

[3] C. Harris, M. Stephens,"A Combined Corner and Edge Detector," In Alvey Vision Conference, 147-151, (1988).

[4] F. Mokhtarian, F. Mohanna, "Performance evaluation of corner detectors using consistency and accuracy measures," Computer Vision and Image Understanding, 102**,** 81–94, (2006).

[5] L.-h. Zou, J. Chen, J. Zhang, L.-h. Dou, "The comparison of two typical corner detection," Second International Symposium on Intelligent Information Technology Application, 211-215, (2008).

[6] W. Wang, R.D. Dony, "Evaluation of image corner detectors for hardware implementation," *Canadian Conference on Electrical and Computer Engineering, 3,* 1285-1288, (2004).

[7] P. Tissainayagam, D. Suter, "Assessing the performance of corner detectors for point feature tracking applications," Image and Vision Computing, 22, 663–679, (2004).

[8] P. Montesinos, V. Gouet, R. Deriche, "Differential Invariants for Color Images," International Conference on Pattern Recognition, 14, 838-840, (1998).

[9] Q. Houben, J.Czyz, J. C. Tocino Diaz, O. Debeir, N. Warzee, "Feature-Based Stereo Vision Using Smart Cameras for Traffic Surveillance," International Conference on Computer Vision Systems, 7, 144-153, (2009).