Софийски университет „Св. Климент Охридски“  
Факултет по математика и информатика

Курсов проект

По Вградени автономни системи  
на тема:

***Използване на вградените системи на мобилните устройства и реализация на алгоритъма за Canny Edge Detector***

изготвили: Цанка Стефанова Енчева ФН 23989  
 Атанас Добромиров Добрев ФН 23990  
Специалност Компютърна графика

16.01.2013

1. Същност на приложението за мобилни устройства Canny Edge Detector

В днешно време повечето от хората притежават модерен мобилен телефон от тип „смартфон“. Той освен за комуникация, предлага много възможности за работа, забавление или развлечение чрез вградените в него системи и софтуерни приложения, написани да използват тези системи.

Приложението за мобилен телефон Canny Edge Detector има за цел да демонстрира възможностите на компютърното зрение, както и вградените в мобилното устройство системи.

Потребителят използва камерата на телефона, за да направи снимка. При вземането на снимка от телефона се използва вградената система Accelerometer, чрез която за определен период от време се определя най-подходящия момент, в който устройството е в покой, за да се направи снимката. За удобство на потребителя чрез използването на вградената система GPS се вземат текущите му координати и се прикачат към снимката. След това тази снимка бива обработена, така че става само от два цвята – черно и бяло. С бяло се изобразяват очертанията на важните обекти, а останалата част от картинката става черна. Чрез използване на различни параметри обектите, които са разпознати като такива, варират.

2. Мотивация - Значение на вградените системи и контурите

Вградените системи в мобилните устройства предоставят на потребителя един лесен и достъпен начин за работа с тях. Едно мобилно устройство комбинира в себе си набор от вградени системи (GPS, камера, Accelerometer, …), което ги прави по-евтини и достъпни за обикновения потребител. Затова за един разработчик е от голямо значение да бъде запознат с техните възможности. Вградените системи в мобилните устройства ни позволяват да отидем още една стъпка по-напред и да обърнем поглед към възможносите на компютърното зрение, което би ни позволило да създадем „умни“ мобилни устройства. Те биха били от голямо удобство например за незрящите хора. За да има компютърно зрение, трябва първо да бъдат разпознати значимите обекти в едно изображение. В този смисъл контурите играят важна роля. Контур(очертание) е рязка промяна в стойностите на сивото в съседство. Значението им за човешката зрителна система е, че осигуряват част от най-значимата информация за изображението. В компютърното зрение те определят очертанията (рамките) на обектите. Освен това дават по-икономично представяне на изображението в сравнение с посочването на стойностите на всички пиксели. Откриването на контурите е първа стъпка от описание на изображението на основата на отделни пиксели (зрение на ниско ниво) към автоматизирано разбиране на съдържанието на изображението (зрение на високо ниво).

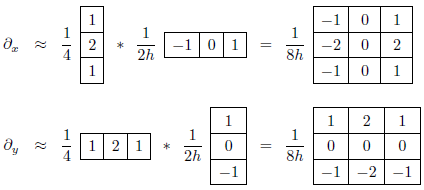
3. Същност на Canny Edge Detector

Canny edge detector е оператор за откриване на контурите на обекти в изображение, използвайки многостъпков алгоритъм. Разработен е от John F. Canny през 1986. Той изготвя доказателство защо тази техника работи. Canny Edge е един от edge detector-ите с най-добро представяне. Целите му са за добро разпознаване на контурите – алгоритъмът трябва да отбележи колкото се може повече реални контури; добра локализация – отбелязаните контури да са възможно най-близо до действителните; контурите да са маркирани само по веднъж и да се избягва маркирането на фалшиви контури от шум. Използва производни от първи ред. Състои се от 3 основни момента, като необходимите параметри са:  
T1,T2 – гранични стойности (прагове)

Основни моменти:

1. Намаляване на шума  
   Тъй като canny edge детекторът се обърква от шума, използваме Гаусов филтър и правим конволюция с Гаусово ядро. Резултатът е леко замъгляване на изображението. При намаляването на шума има 2 параметъра, които могат да бъдат променяни – колко пъти да се направи конволюцията (в брой стъпки) и sigma, който отговаря на стандартното отклонение на Гаусовото разпределение

void presmooth (float \*f, float sigma, int steps, int width, int height)  
Граничните пиксели по ръба на изображението ги взимаме огледални на тези до тях.

1. Намиране на приближена стойност на наклона(градиента)   
   Нека *u* е изображението след стъпка 1. Очертанията на един обект могат да сочат в различни посоки. Canny алгоритъмът използва филтъра за откриването на вертикални, хоризонтални и диагонални очертания. Трябва да пресметнем първата производна в x и y направленията. За целта въвеждаме *nabla* оператора (градиент):   
   C:\Users\CECI\Desktop\nabla-op.PNG  
   Операторът nabla може да се използва като вектор, т.е. за функция *f(x,y)* получаваме:  
   C:\Users\CECI\Desktop\nablaf-op.PNG  
   Пресмятаме големината на градиента:  
   C:\Users\CECI\Desktop\grad-magnitude.PNG  
   Самите производни преди това сме изчислили с приближение с оператора на Собел, използвайки следните маски за пикселите:  
   

getderivatives(image u, dx, dy, width, height)

Изчисляваме величината|∇u| и ъгъла на ориентация φ = arg(|∇u|) на |∇u| (използвайки полярни координати) φ = arctan(∂y / ∂x)  
getdirection(dx, dy)  
Ъгълът се приближава до някой от четири основни ъгъла за вертикала, хоризонтала и двата диагонала(0, 45, 90, 135 градуса например)   
Кандидатите да участват в контура са местата, където |∇u| надвишава прага T1.

1. Nonmaxima suppression  
   *Цел*: изтъняване на очертанията  
   За всеки пиксел, кандидат да участва в контура, вземаме посоката в грида от пиксели (от 4 посоки), която е „най-ортогонална“ на него. Ако в тази посока някой от 2та съседни пиксела има по-голям наклон (градиент), маркираме средния пиксел и го отбелязваме като черен.

nonmaxima\_suppression(image u, dx, dy, width, height)  
apply\_nms(x1, x2, \*x2, x3)

1. Ограничение от двете страни  
   *Цел*: Да вземем само контурите, които ни интересуват  
   Правим предположението, че важните контури са по продължението на непрекъснати линии – така можем да отхвърлим някои пиксели от шум, които не са контури, а имат голям градиент.  
   Първо, прилагаме по големия праг – T2. Използваме точки над втория праг T2 като seed (за пресяване). Използвайки информацията за ориентацията на контурите, проследяваме контурите и прилагаме по-ниския праг. Добавяме всички съседи, които са над по-ниския праг Т1 и за тях прилагаме рекурсивно същия алгоритъм.  
   hysteresis\_thresholding(image u, T1, T2, width, height)  
   traceedge(xi, yj, image u, T1, T2)

4.Демонстрация и ръководство

Приложението може да се използва с предварително зададени изображения , които са подходящо избрани, за да се покажат основните моменти при Canny Edge Detection. Също така приложението предоставя възможност за снимане избрани от потребителя обекти чрез следния интерфейс:



След като заснемем подходящо изображение, приложението предоставя следния интерфейс за работа с Canny Edge Detector.



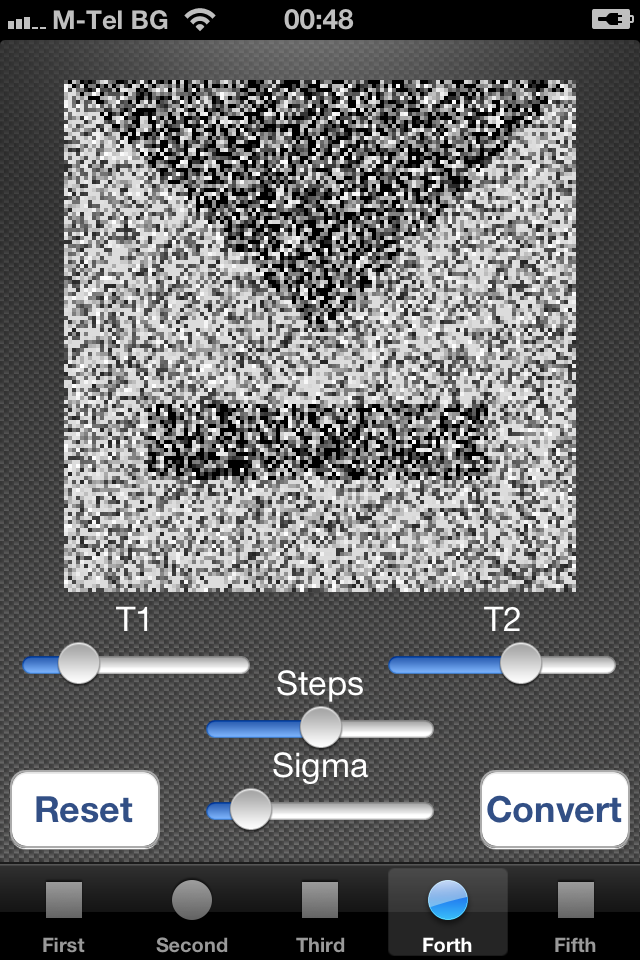
Слайдърите T1 и T2 се използват за избиране на двата threshold параметъра. Steps се използва за избиране на броя стъпки, които ще се приложи замъгляването. Sigma се използва за определяне на стандартното отклонение на Гаусовото разпределение.

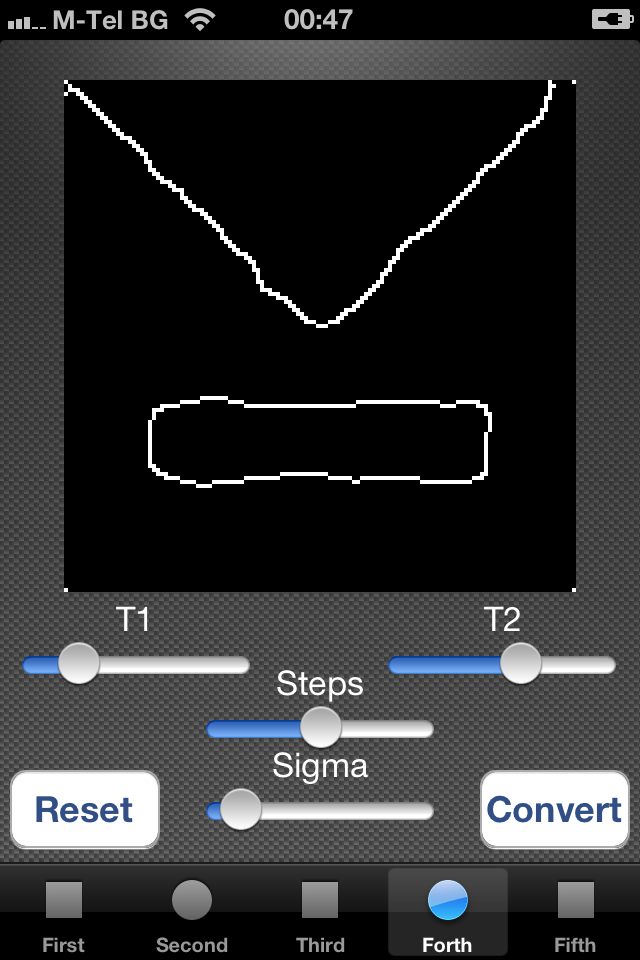
Бутонът Convert прилага Canny Edge Detector алогоритъма върху изображението. Резултатът от него може да се види на следващото изображение:

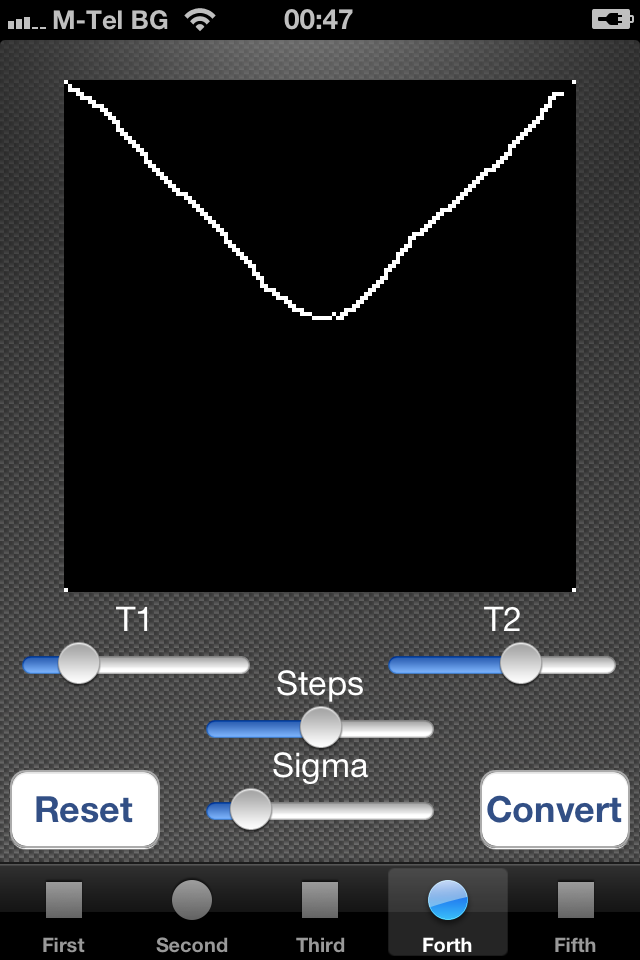


Бутонът Reset връща към първоначалното изображение.

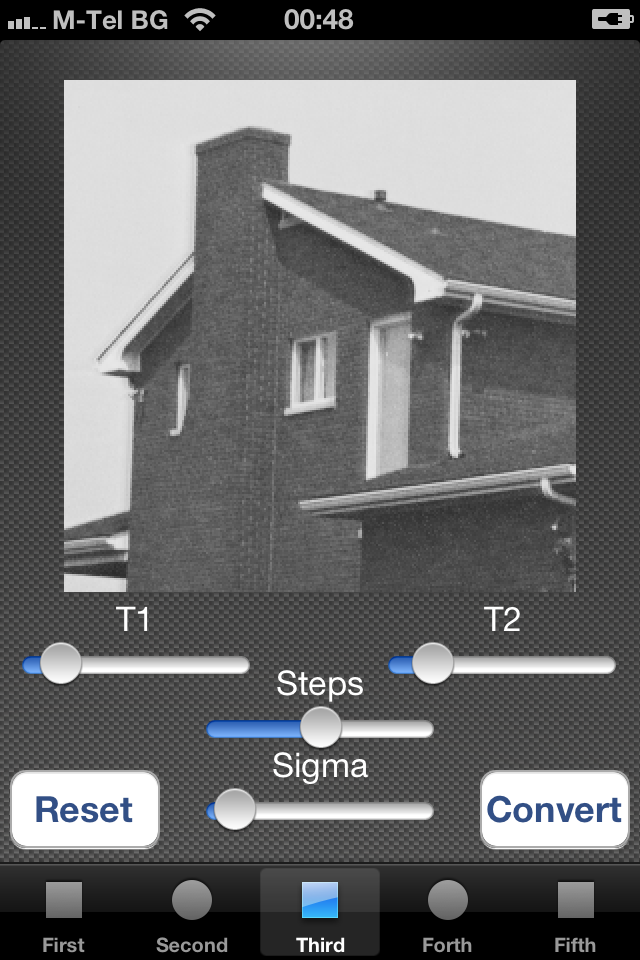
На следващите 3 изображения можем да видим разликата в използването на различен параметър sigma.







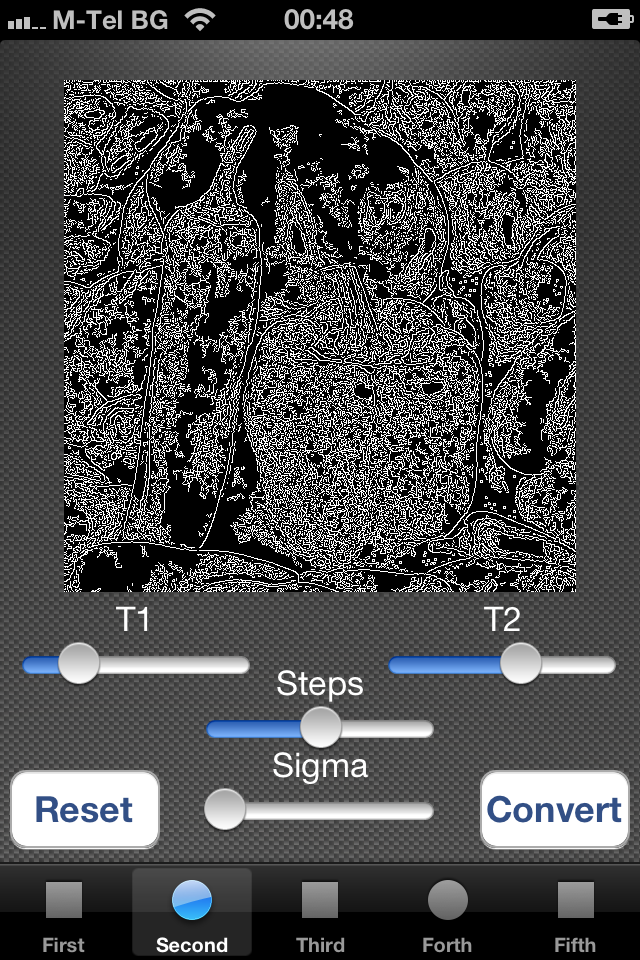
На следващите 6 изображения можем да видим промените, които настъпват, когато променяме T1, T2 и sigma.





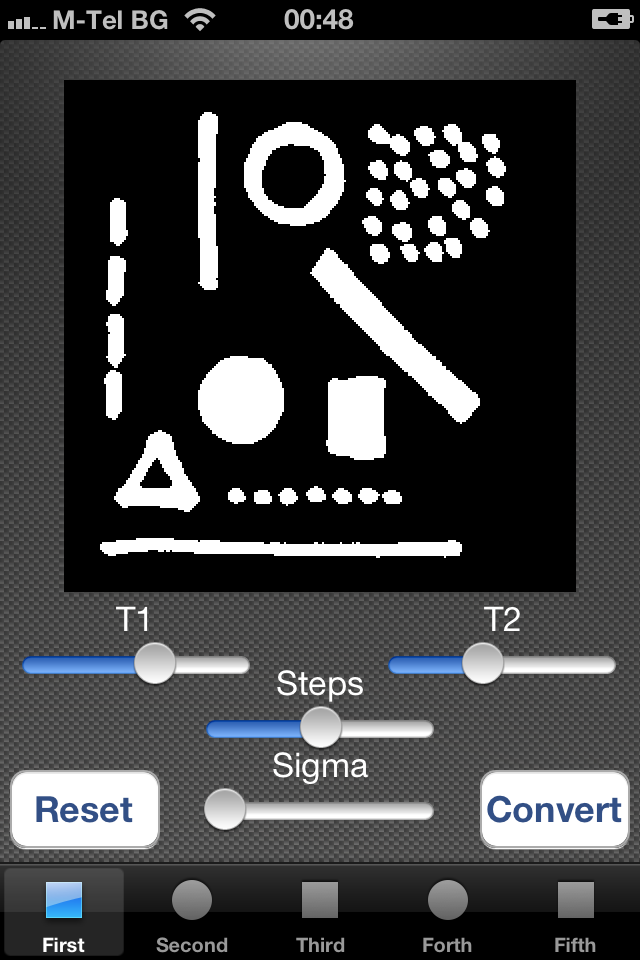


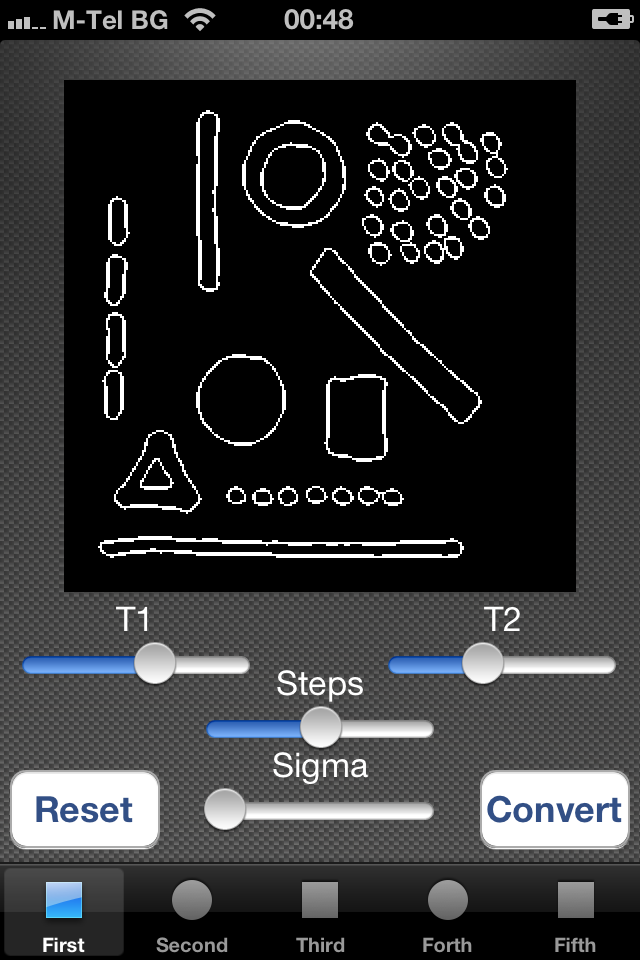






На следващите 2 изображения виждаме случай, в който sigma не оказва влияние, тъй като няма шум в изображението.





Технологии:  
Xcode for MacOSX  
iOS for iPhone  
Objective C, C

Източници:  
http://en.wikipedia.org/wiki/Canny\_edge\_detector  
http://www.aishack.in/2011/06/the-canny-edge-detector/  
https://developer.apple.com