Babes-Bolyai Tudományegyetem
Matematika-Informatika Kar

Tárgykövető robot

Máté István-Zoltán
Vezetőtanárok : Soós Anna, Csató Lehel

Kolozsvár
2006

1.1. Bevezető:

A projekt célja egy számítógép irányítású, két léptetőmotorbol álló, gömkordinátában mozgó, kamerával felszerelt robot készítése, amely képes bizonyos tulajdonságú tárgyak követésére, egy kamera mozgatásával. A léptőmotrok a párhuzamos porton *(Parallel Port)* keresztül kommunikálnak a számítógéppel, a kamera pedig az USB *(Universal Serial Bus)* porton keresztül. A tárgy felismerése szín alapján RGB *(Red Green Blue)* színmezőben történik. A tárgy felísmerése különböző módszerek segítségével, színsűrüség, illetve színhisztogram segítségével történik. A tárgy követése kanonikus, illetve Kalman filterrel valósul meg. A program implementálása Microsoft Visual Studio .Net 2003 alatt történt, C# programozási nyelvben.

1.2. Az eszközök leírása:

Az alábbiakban azok az eszközök *(hardware)*kerülnek leírásra amelyekből felépül a robot, valamint a robot használ.

1.2.1. A párhuzamos port

A számítástechnikában a párhuzamos portot *(Parallel Port)* interfésznek tekintik a számítógépes rendszer, ahol az a adatok párhuzamosan vannak közvetítve a különböző egységek közt, és egy külső hardware egység *(nyomtató, egér)* közt. A párhuzamos port egy bit-et (Bit), közvetít minden egyes szálon *(Pin)* egy időegység alatt, és nem többet egy adott időegységen belül, mint például a soros port *(Serial Port)*. A porton az adatátviteli bitteken kívül található pár jelzőbit, amelyek célja jelezni, hogy az eszköz mikor áll készen a müködtetésre, legyen az a hardware eszköz, vagy maga a számítógép. Manapság az USB *(Universal Serial Bus)* felváltotta, a párhuzamos port használatát, és sok ma gyártott számítógépen *(2006)* már mellőzik a párhuzamos portot, pénzsporlás céljából, és azért mert használata elavultnak tekintett. A soros port müködtetése hasznosabbnak bizonyult kissebb mérete miatt, és nagyobb adatátvili képességei miatt.

1.2.2. A léptetőmotrtok:

A léptetőmotor ahogy neve is mondja, egy olyan motor amely egy bizonyos lépésszöggel képes fordulni jobb, illetve bal irányba. A léptetőmotor abban különbözik a hagyományos motortól, hogy ez kis lépésekben képes fordulni, mig egy hagyományos motor fordulatait percenként mérik *(fordulat/perc)*,

nem képes "apró" lépésekre. A léptetőmotrok TEAC 14769070-10 típusu 5.25" lemezmeghajóból *(Floppy Drive)* származó motrok. A motrok 12V-al *(Volt)* működnek. A motrok vezérlésével egy interfész foglalkozik, amely a párhuzamos portról érkező jelt átkapcsolja 12V-os feszültségre a motrok működtetésére. A motrok vezérlése egy 5 szálas kábelen keresztül történik, amelyből 1 földelés, és a többi 4 lehetővé teszi a motor bizonyos irányba való elmozdulását.

1.2.3. Az interfész:

Az interfész egy olyan elektrónikus szerkezet, amely a párhuzamos portról érkező jelt a motrok mozgatására alkalmas jellé alakítja. Az interfész alapjában véve kapcsolóként müködik, amely a párhuzamos porton érkező 3.8 V-os feszültséget 12V-ra kapcsolja át. Felépítesében motronként 4 tranzisztor, 4 ellenállás, és 4 dióda szerepel. Az ellenállás szabályozza a tranzisztorhoz érkező feszültséget, a tranzisztor a párhuzamos portról érkező feszültségkor nyit, és a 12V-os feszültséget átadja a motornak.

1.2.4. A kamera:

A projektben használt kamera, USB portra csatlakozó , Logitech Quickcam Express. A kamera 352x288 pixeles *(Pixel: színkomponens alapaegysége a számítógép képernyőjén)*, felbontásban képes képeket közvetíteni, és 15 fps *(Frame Per Second)* a frissítése videó felvétel készítésekor. Egy lényeges probléma a kamerával, hogy mivel kis felbontású képet közvetít, és egy olcsó kameratípus teljesítménye nagyon függ a fényviszonyoktól. Nappali fény használatakor az érkezett kép jó minőségű, viszont gyenge fényviszonyok esetén a képen nagy lesz a zaj, ez gyakorlatilag abban nyilvánul meg, hogy a kép "bolházik".

1.2.5. A robot

A robot motrai úgy vannak elhelyezve, hogy lehetövé tegyék a kamera gömbkoordinátában való mozgását. Az első motor a rotorjával *(Rotor a motorban mozgó vasmag)* merőlegesen van elhelyezve a föld síkjára, a második motor viszont ennek a rotorjára van merőlegesen elhelyezve, és ennek a rotorján található a kamera.

1.3. A tárgy meghatározása

1.3.1. Bevezető:

A tárgyak meghatározása a mesterséges intelligencia témakörében a '60-as évektől kezdődöen jelentett problémát, amikor a számítógépek fejlődése lehetővé tette az effajta kérdések felvetődését, de egésszen a '80-as évekig nem lehetett komolyan foglalkozni evvel a témával, mert a számítógépek nem tették lehetővé a képfeldolgozást. A tárgyak felismerése, és követése a '90-es évektől kezdődöen kihívást jelentett, viszont a hétköznapi számítógépeken történő követés nem volt lehetséges, bár a tárgyfelismerés már lehetséges volt statikus képek esetén. A valós időben történő feldolgozás nehézségei miatt, ami elsősorban a nagy mennyiségű adatfeldolgozást jelentett valós időben, sokáig nem volt lehetséges evvel a témakörrel foglalkozni. Különálló hardware egységek létrehozásával már lehetett számítógépes tárgykövetést megvalósítani a '80-as évektől, de ez általában csak a nagy cégek, és vállalatok esetében volt lehetséges, mivel egy hardware egység létrehozása rengeteg pénzt és mérnöki munkát igényel. Az első tárgykövetési alkalmazások az amerikai katonaságtól származnak, tárgykövető rakéták alkalmazásaiban fordultak elő a '80-as évek közepétől. A számítógépek egyre nagyobb ütemben való fejlődese lehetővé tette, hogy lehetséges legyen a tárgykövetés hétköznapi felhasználóknak is. Tárgyfelismerés terén rengeteg algoritmus létezik, egy részük a tárgy alakján alapuló tárgyfelismerést alkalmaz. Ez a módszer hátránya a nagy számítási adat feldolgozása, es az algoritmusok nagy szamítasi igényei, ami az alkalmazás lelassulásához vezet. Ezeket a módszereket általában több számítógépes rendszereken alkalmazzák valós időben való követésre, így érik el a megfelelő teljesítményt. Ezek a módszerek előnye, hogy a tárgyat egyértelmüen meghatározzák. Ezt a módszert általában fekete-fehér kamerák esetében alkalmazzák, mert kevessebb számítást kell végezni a tárgy meghatározásához, és mivel nem lehet másképp egyértelmüen meghatározni a tárgyakat fekete-fehér színmezőben. Egy másik módszer, amelyet színes kamerás alkalmazások esetében használnak szìnsűruségen alapuló módszerek, de ezen kívül rengeteg más típusú algoritmus van. Az általam alkalmazott módszer alapja a tárgy színhisztogramjának elkészítése.

1.3.2. Színhistogram alapú tárgyfelismerés

A számítógépes színábrázolás alapegysége a pixel. Egy pixel három komponensből áll: R G B *(Red, Green, Blue)*, azaz piros, zöld, kék. A számítógép képernyőjén megjelenő képek ez a három szín keveréséből állnak elő. Az RGB színmező komponenseinek terjedelme 0-255 van, egy szín értéke minnél

inkább tart a 0-hoz, anál inkább sötétebb, tehát a fekete szín R=0, G=0, B=0, és a fehér R=255, G=255, B=255. A Logitech Quickam Express kameráról a képek RGB felépítésben, 24 bittes színábrázolásban érkeznek. A hisztogram alapú targyfelismerés lényege, hogy az adott térrészben található pixeleknek figyeli a gyakoriságát színkomponensenként.

1.3.3. A tárgy megadása, és a hisztogram előkészítése:

A színskálát felosztjuk k egyenlő részre.

$$xR_i = [0..x_i], x_i = 0 + 255/i, i = 0..k$$

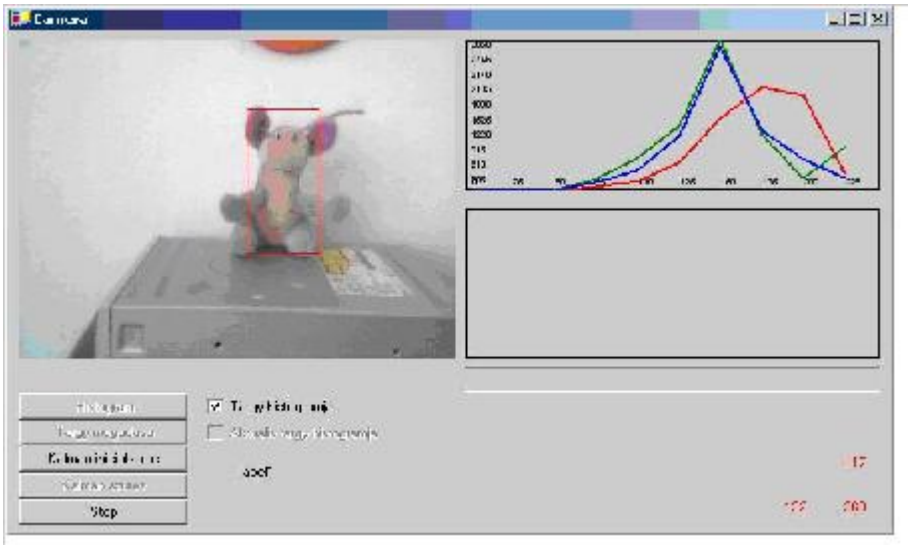
$$xG_i = [0..x_i], x_i = 0 + 255/i, i = 0..k$$

$$xB_i = [0..x_i], x_i = 0 + 255/i, i = 0..k$$

A tárgy megadása egy téglalapban történik. Ebből a téglalapból kiindúlva a program végigfutja minden sorát, és menézi, hogy az adott szín melyik intervallumba tartozik, és növeli az adott intervallumhoz tartozó, értéket, amely az előfordulást figyeli. Lekérdezzül az aktuális pixel értékét, és meghatározzuk ColorR, ColorG,ColorB-t.

X_l, X_r az aktuális tárgy x koordinátái, Y_l, Y_r pedig az y koordinátái.

x_i a tárgy aktuális x koordinátája , ahol $i \in [X_l..X_r]$
y_i a tárgy aktuális y koordinátája, ahol $j \in [Y_l..Y_r]$
Color = Color(x_i ,y_i)
$Color_R$ az aktuális szín R komponense
$Color_G$ az aktuális szín G komponense
$Color_B$ az aktuális szín B komponense
Ha $Color_R \in xR_i$ akkor elofordulR[i] = elofordulR[i]+1
Ha $Color_G \in xG_i$ akkor elofordulG[i] = elofordulG[i]+1
Ha $Color_B \in xB_i$ akkor elofordulB[i] = elofordulB[i]+1
Ezt az eljárást végigfuttatjuk minden (x_i ,y_i) -re.
Az így megkapott elofordulR, elofordulG, elofordulB fogják megadni az adott tárgy hisztogramját.
A programban a megkapott értékeket nem hisztogramként ábrázoljuk, hanem gyakorisági görbeként.



1.1. ábra. A gyakorisagi görbék

A piros görbe a tárgy piros színkomponensének a gyakorisági görbéje, a zöld görbe a zöld komponens görbéje, és a kék a kék komponens gyakorisági görbéje. A hisztogram készítése a piros téglalapban történik.

1.3.4. Hibaforrások:

Elsődleges hibaforrás a fényviszonyok változása, például egy árnyék rávetülése a képre, ekkor a tárgy hisztogramja olyan módosulásokat szenved, amelyek nem garantálják a tárgy optimális felismerését. Ennek kiküszöbölését úgy próbáljuk , hogy a lehetséges tárgytalálatot nem a hisztogrammal való tökéletes egyezés esetén adjuk meg, hanem a lehetséges tárgytalálatok minimumával. A minimum számolása az eredeti tárgy hisztogramjához képest történik.
Mivel a hisztogram meghatározása $O(n^2)$ bonyolultságú, tehát implementáláskor két for *(Amíg)* ciklust tartalmaz, végrehajtásuk rendkívül időigényes. Emiatt relatív kis téglalapban végezhető el a hisztogram meghatározás, és a követés. Egy 352x288 pixeles kép teljes befutásához 101376 műveletet kell elvégeznie a számítógépnek, viszont nincs hozzászámolva, hogy ezek a számítások mögött hány számítást végez el ténylegesen a program és az operációs rendszer, tehát ez csak egy relatív értéket jelől.

1.4. A tárgy követése, és lehetséges poziciójának meghatározása

Egy mozgást végző tárgy lehetséges poziciójának meghatározására alkalmas egyszerű és jól müködő módszer a Kalman szűrés *(Kalman Filter)*. A projektben ez a tárgykövetési módszer van alkalmazva.
Ez a módszer csak egy bizonyos valószínűséggel adja meg a tárgy helyzetét, ezt, és a hasonló módszereket "jós" módszereknek nevezik *(prediction)*, mivel gyakorlatilag megjósolják a tárgy következő időpillanatbeli a helyzetét. Ezeknek a módszereknek az előnye, hogy egy lineáris egyenlet megoldásával relatív gyorsan meg lehet adni a tárgy helyzetét.

1.5. A Kalman szűrő leírása:

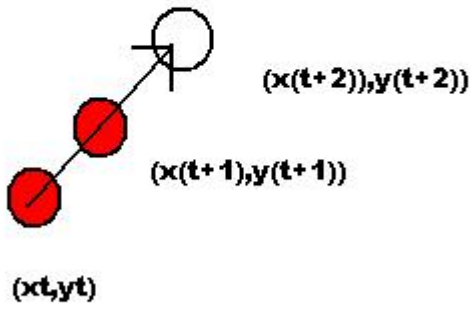
1.5.1. Bevezető:

A szűrő neve a szűrő feltalálójától származik, vagyis Rudolf E. Kalmantól, bár Peter Swerling kidolgozott már egy hasonló szűrőt korábban.
Stanley Schmidtet ismerjük mint az első ember aki alkalmazta a Kalman szűrőt. Ez 1958-ban történt amikor Stanley lehetőséget kapott arra, hogy az Apollo programon belül alkalmazást nyerjen. A szűrő papíron Swerling által volt kifejlesztve(1958), majd Kalman által (1960), és Kalman és Bucy által (1961).
A Kalman szűrőnek egy nagy sokaságát fejlesztették ki napjainkig, az eredeti Kalman szűrőbúl, amelyet ma egyszerű Kalman szűrőnek hívunk.
Manapság a Kalman szűrőt használják a távközlésben, és gyakorlatilag minden bonyolultabb elektronikai eszközben.

1.5.2. A Kalman szűrő müködése

A Kalman szűrő becslést próbál adni egy tárgy következő helyzetéről egy $t+n$ időpillanatban, ha a $t+(n-1)$. időpillanatban ismert a poziciója. Ahogy az ábrán is látható, az (x_t, y_t) helyzetből a tárgy (x_{t+1},y_{t+1}) helyzetbe mozdul el 1 időpillanat alatt. A Kalman szűrő lényege, hogy becslést ad a következő pillanbeli pozicióra, és amennyiben a tárgy lineáris mozgást végez akkor az ($x_{t+1}-x_t$, $y_{t+1}-y_t$) vektor által megadott irányba fog elmozdulni, amint az alábbi ábrán látható.

Az egyszerű Kalman szűrő tehát egy lineáris becslést ad a tárgy $t+n$. időpillanatbeli helyzetéről .



1.2. ábra. A Kalman szűrő müködése

x_{t+n} a tárgy x koordinátája a $t+n.$ időpillanatban.
y_{t+n} a tárgy y koordinátája a $t+n.$ időpillanatban.
v_n a tárgy sebessége a $t+n.$ időpillanatban.
$a_{n)}$ a tárgy gyorsulása a $t+n.$ időpillanatban

A Kalman szűrő tehát egy rekurzív algoritmus, ami lineáris becslést ad a tárgy poziciójáról egy $t+n.$ időpillanatban, ismerve a tárgy pozicióját a t. időpillanatban.

Ekkor a mozgásegyenlet:

$$\mathrm{x_{(n+1)*T}} = \mathrm{x_{n*T}} + \mathrm{T} * \mathrm{v_{n*T}} + \frac{\mathrm{T}^2}{2} * a_{n*T}$$

Mivel ha T ,a két időpillanat közti különbség, abban az esetben, ha elég kicsi, tehát nagyon rövid időközönként figyeljük a tárgy mozgását, akkor a

$$\mathrm{T} * \mathrm{v_{n*T}} + \frac{\mathrm{T}^2}{2} * a_{n*T}$$

összeg nullának tekinthető.
Tehát:

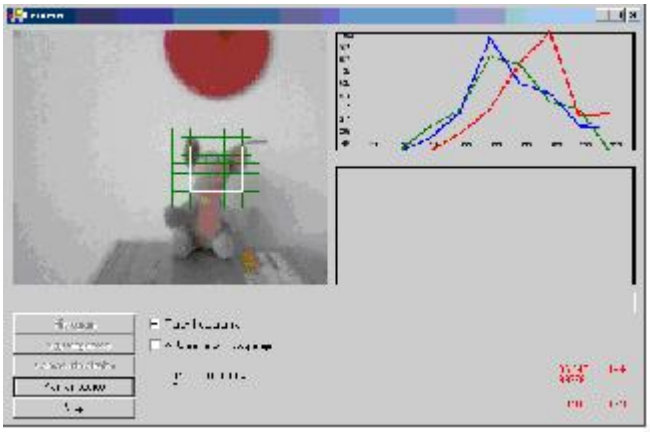
$$\begin{bmatrix} \mathrm{x_{k+1}} \\ y_{k+1} \\ dx_{k+1} \\ dy_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x_k \\ y_k \\ dx_k \\ dy_k \end{bmatrix} + w_k$$

ahol w_k a tárgy pályájának korrekciója.

1.5.3. A Kalman szűrő által meghatározott pálya korrekciója

Mivel a Kalman szűrő nem adja meg taljes pontossággal a tárgy pályáját, szükség van egy korrekcióra. Ezt a korrekciót pedig úgy lehet elvégezni, ha a meghatározott pálya egy környezetében vizsgáljuk a tárgyat. A tárgy vizsgálata bizonyos szűrő segítségével történik, jelen esetben a színhisztogram segítségével.

A fenti ábrán látható piros téglalap adja meg a tárgy aktuális pozicióját, amit a Kalman szűrő határozott meg, a fehér téglalap pedig megadja a korrekciót, amit a piros téglalap környezetében találtunk. A téglalap környezetében a keresést a zöld egymásraépülő téglalapok adják, ezek közül történik a minimum meghatározás. A pálya korrekciója biztosítja azt, hogy a Kalman szűrő ne szaladjon el egy bizonyos irányba, és stabilizálja azt, biztosítva a program



1.3. ábra. Pálya korrekciója

optimális müködését.

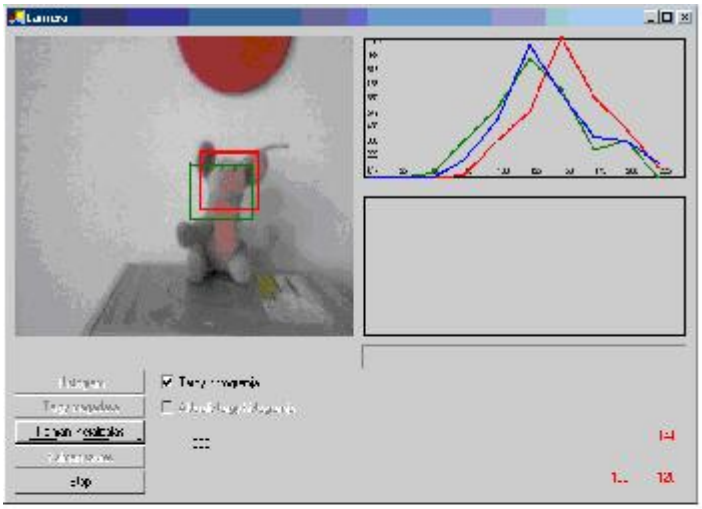
1.5.4. A Kalman szűrő inicializálása:

A Kalman szűrőt inicializálni kell, mivel a szűrő a tárgy időbeli helyzetével dolgozik, kezdetben $t = 0$,és nem ismert a tárgy pozíciója. Kezdeti pillanatban a x_k egyenlő lesz a tárgy kezdeti pozíciójával, a következő számítási pillanatra pedig, meg kell adjunk egy irányt, amelyre vélhetően a tárgy el fog mozdulni.

A képen látható piros téglalap megadja a tárgy kezdeti pozícióját, a zöld téglalap pedig azt a pozíciót, amelybe vélhetőleg el fog mozdulni. Ezek után a Kalman szűrő kiszámolja az elmozdulásokat mindkét irányban: dx_t -t, illetve dy_t -t. A továbbiakban a Kalman szűrú valamit a korrekció biztosítja a tárgy következő pillanatbeli helyzetét.

1.6. Megjegyzések a robot képességéről és problémáiról

Amint a fentiekből láthatjuk a robot müködtetésére két erős módszert használunk, a színhisztogram meghatározást, valamint a Kalman szűrőt.
A robot általában problémák nélkül meghatározza a tárgyat, viszont érzékeny a fényviszonyok változására, mint például, árnyék esik az adott tárgyra, fényvisszaverés a tárgyról, ami hirtelen megváltoztatja a keresési körülményeket, ezért a tárgy megtalálása nem biztosított, viszont egy elég jó becslés nyerhető



1.4. ábra. A Kalman szűrő inicializálása

annak poziciójának a meghatározásáról.
Másrészt problémák akkor következnek be, amikor a tárgy kilép az adott képből, tehát a motrok mozgatásával kell keresnünk azt. A motor egy lépésben relatív kicsit lép, ez az érték körülbelül 50 pixeles értéknek felel meg, ha ebben a pillanatban gyorsan mozog a tárgy, akkor könnyen elvesztheti a fókuszt, illetve ha túl lassan mozog, akkor az aktualis poziciónál kissebbet határoz meg. Mivel színhisztogramos felísmerést végzünk, nem kereshhetünk teljes 352x288-as képen, hanem csak a tárgy egy környezetében. Ha ezt a környezetet nagyon kiterjesztjük, akkor lassul a program, ennek eredményeként a képfeldolgozás, a tárgy megint eltűnhet, annélkül, hogy követni tudnánk. Tehát nehéz pontosan meghatározni azt a sebességet amellyel mozgatni kellene a tárgyat képváltáskor, illetve a számítógép túlterhelése miatt. Amennyiben a számítógép nagyobb teljesítményű számításelvégzésre lenne képes, úgy jobb lenne a tárgymeghatározás is.

1.7. Könyvészet

1. Real-Time Kernel-Based Tracking in Joint Feature-Spatial Spaces - Changjiang Yang, Ramani Duraiswami, Ahmed Elgammal and Larry Davis cs-tr.pdf

2. Object Tracking and Kalman Filtering - Hai Tao lec15.pdf

3. Kalman filter for vision tracking - Erik Cuevas, Daniel Zaldivar and Raul Rojas tr-b-05.pdf

4. Bayesian Filtering and Integral Image for Visual Tracking - Bohyung Han Changjiang, Yang Ramani Duraiswami, Larry Davis wiamis2005.pdf

5. Object Recognition with Color Cooccurrence Histograms - Peng Chang,John Krumm - Microsoft Research ColorConcurrenceCVPR.pdf