Corso di Reti ad hoc e di sensori
Progetto finale

Andrea Maggiordomo – mggndr89[at]gmail.com
Laurea Magistrale in Informatica, Università di Pisa

2015

1 Introduzione

Scopo del progetto è la realizzazione di un sistema che utilizzi una rete di telecamere per individuare oggetti in movimento e calcolarne le relative coordinate nello spazio. L'applicazione per cui è stato pensato è l'estrazione automatica delle coordinate dei dischi che si muovono sul tavolo ad aria presente nel laboratorio di Fisica dell'Università di Pisa, ma un sistema di questo tipo può essere facilmente esteso ad applicazioni di monitoraggio/videosorveglianza. L'applicazione specifica per cui il sistema è stato sviluppato ha permesso di fare alcune (forti) assunzioni sull'ambiente monitorato dalle telecamere: in particolare, si assume che tutti gli elementi rilevati si muovano sulla stessa superficie, e che le telecamere siano in posizione fissata.

2 Componenti del sistema

I dispositivi utilizzati per il monitoraggio della scena sono dei Raspberry-PI equipaggiati con *Camera module* e l'approccio utilizzato è una versione semplificata di quanto descritto in [3]. Ciascun Raspberry ospita un server sequenziale che permette ad un client di attivare la telecamera e ricevere in streaming i dati elaborati dal sensore. Vista la bassa potenza di calcolo dei dispositivi, viene impiegato un cluster di telecamere per incrementare la frequenza di campionamento della scena e tracciare in maniera più affidabile gli oggetti in movimento.

L'elaborazione delle immagini è effettuata utilizzando la libreria OpenCV (`http://opencv.org`), mentre per interfacciarsi con il Camera module del Raspberry la libreria usata è RaspiCam C++ (`http://www.uco.es/investiga/grupos/ava/node/40`).

3 Caratteristiche del sistema

3.1 Rilevamento degli oggetti

Gli oggetti in movimento sono rilevati utilizzando un algoritmo che opera per sottrazione di background. Questa è una scelta ragionevole nello scenario considerato (le telecamere sono in posizione fissata e gli oggetti da individuare sono in costante movimento); inoltre, i parametri dell'algoritmo sono relativamente semplici da calibrare.

3.2 Coordinate della superficie

Per convertire le coordinate di un oggetto rilevato in un fotogramma nel sistema di riferimento relativo alla superficie su cui si muove, un'omografia tra il piano dell'immagine e la superficie stessa viene calcolata in fase di configurazione, specificando quattro punti di controllo visti da ciascuna telecamera e generando una trasformazione prospettica [1]. (Al momento l'implementazione prevede che questi punti descrivano un rettangolo che abbia un vertice coincidente con l'"origine" della superficie; questo non è strettamente necessario per calcolare la trasformazione, i quattro punti di controllo potrebbero avere coordinate arbitrarie). Una volta calcolata l'omografia, questa viene utilizzata localmente per eseguire le conversioni ogni volta che il sensore viene attivato.

3.3 Identificazione e tracciamento dei singoli oggetti

I dati ottenuti da ciascun sensore sono convogliati nel client, che li utilizza per identificare e tracciare gli oggetti. A ciascun oggetto viene associato un filtro di Kalman [5] che ne modella lo stato, con lo scopo di prevederne la posizione ai fotogrammi successivi: quando i dati relativi ad un nuovo fotogramma sono ricevuti, ad ogni oggetto tracciato è associato il rilevamento più probabile risolvendo un problema di assegnamento tra le posizioni previste da ciascun filtro di Kalman e quelle rilevate nel nuovo fotogramma. Informalmente, se P è l'insieme delle posizioni previste dai filtri al frame corrente, D l'insieme delle posizioni rilevate al frame corrente e $c_{ij} = \|p_i - d_j\|$ il costo di assegnare la previsione $p_i \in P$ al rilevamento $d_j \in D$, si cerca l'assegnamento di ogni elemento di P con un (diverso) elemento di D che minimizza il costo complessivo degli accoppiamenti.

Questo problema è stato risolto con l'algoritmo ungherese [2, 4]. Il codice è adattato dall'implementazione di Mattias Andrée disponibile all'indirizzo `http://github.com/maandree/hungarian-algorithm-n3`.

3.3.1 Specifica del filtro di Kalman

Lo stato di un oggetto è definito come la sua posizione e la velocità con cui si muove sulla superficie:

$$x = [x, y, v_x, v_y]^T$$

Il filtro non contempla nessun tipo di input esterno; l'equazione che definisce l'evoluzione del sistema diventa pertanto:

$$x_k = Ax_{k-1} + w_{k-1}$$

La matrice di transizione viene aggiornata ad ogni iterazione, specificando l'intervallo di tempo tra il frame elaborato ed il precedente:

$$A = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Le misurazioni riguardano unicamente la posizione dell'oggetto; in quest'ottica le variazioni di velocità sono modellate come rumore.

4 Esecuzione

I paragrafi seguenti descrivono come configurare il sistema ed eseguire un esperimento, assumendo che esista già una rete alla quale tutti i dispositivi sono collegati.

4.1 Server

Una volta compilato il sistema e posizionata la telecamera, occorre procedere con la configurazione del server che permette di calcolare l'omografia utilizzata dal server per convertire le coordinate degli oggetti rilevati. Questo passaggio avviene per mezzo di un'interfaccia grafica, si consiglia pertanto collegarsi in `ssh` al dispositivo utilizzando lo switch `-X` per abilitare il forwarding di X11 e visualizzare la finestra sul proprio terminale. Il server viene lanciato in modalità di configurazione specificando lo switch `-c` e passando come ulteriori parametri la larghezza e l'altezza della regione monitorata. Per esempio, per rilevare movimenti su un rettangolo che misura 975×1300 millimetri:

```
./CameraServer -c 975 1300
```

Questo apre una finestra nella quale è possibile specificare i quattro vertici del rettangolo, come mostrato in figura 1.

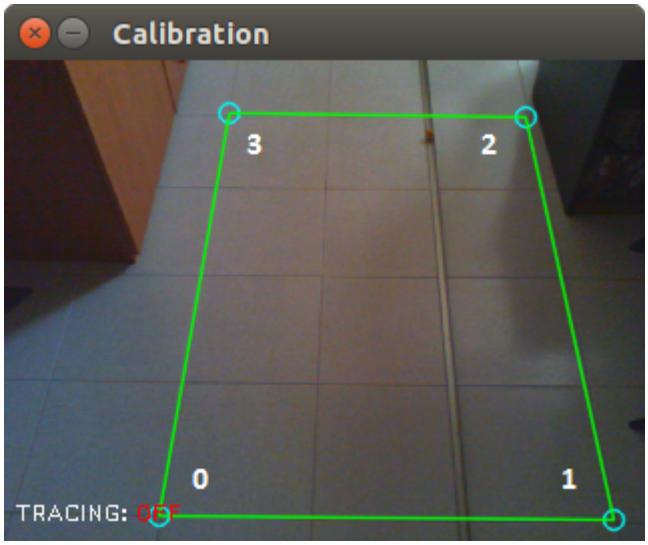

Figura 1: Definizione dei punti di controllo per calcolare la trasformazione prospettica.

Una volta completata la configurazione, il server va avviato e rimane in attesa di connessioni sulla porta `12345`.

Alcuni parametri della telecamera (come esposizione, contrasto, saturazione etc...) e dell'algoritmo di background subtraction possono essere configurati modificando il file `params/camera_params.yaml`. Per ulteriori dettagli si rimanda ai commenti presenti nel file `calibration.hpp`.

4.2 Client

Quando il client viene lanciato occorre specificare quattro parametri da riga di comando; i primi due parametri indicano le dimensioni del rettangolo entro il quale si vuole visualizzare informazioni sull'attività rilevata (probabilmente uguali ai parametri passati al server in fase di configurazione), mentre i successivi parametri definiscono la risoluzione della finestra nella quale il client visualizza l'attività rilevata dai server. Ad esempio, per visualizzare i dati in una finestra di 730×975 pixel:

```
./Client 975 1300 730 975
```

Durante l'esecuzione è possibile impartire al client i seguenti comandi:

`connect address [port]`
: Si collega all'indirizzo specificato, aggiungendo una nuova entry alla lista delle connessioni attive. Il comando può essere ripetuto per connettersi a più server. La porta utilizzata di default è `12345`.

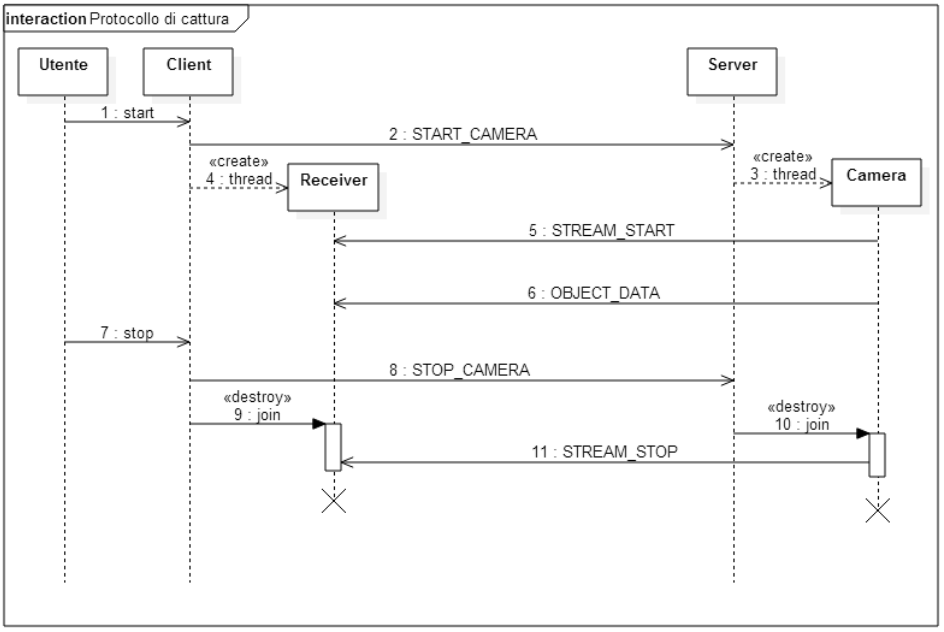

Figura 2: Sequenza di avvio e arresto del monitoraggio.

`start`
: Invia ad ogni server in ascolto il comando per attivare i sensori ed iniziare una cattura.

`stop`
: Arresta i sensori ed interrompe la cattura.

`close`
: Chiude il client.

4.3 Protocollo di comunicazione

Il protocollo di comunicazione per attivare e disattivare un sensore è descritto in figura 2. Quando un utente decide di avviare la cattura, il client crea un thread ricevitore ed invia al server un messaggio di tipo `START_CAMERA`, al quale il server reagisce creando un nuovo thread che si interfaccia con la telecamera. Questo thread notifica al ricevitore l'inizio della cattura inviando un messaggio `STREAM_START`, seguito da una serie di messaggi di tipo `OBJECT_DATA` (uno per ogni fotogramma processato) che contengono le posizioni degli oggetti individuati e un timestamp con il tempo di cattura. Quando il sensore viene arrestato, il server invia il messaggio `STREAM_STOP` per notificare la chiusura del flusso di dati.

Riferimenti bibliografici

[1] Heckbert P. S. (1989). Fundamentals of texture mapping and image warping. Master thesis. *University of California, Berkeley.*

[2] Kuhn H. W. (1955). The hungarian method for the assignment problem. *Naval research logistics quarterly*, **2**(1-2), 83–97.

[3] Medeiros H.; Park J.; Kak A. C. (2008). Distributed object tracking using a cluster-based kalman filter in wireless camera networks. *Selected Topics in Signal Processing, IEEE Journal of*, **2**(4), 448–463.

[4] Munkres J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial & Applied Mathematics*, **5**(1), 32–38.

[5] Welch G.; Bishop G. (2006). An introduction to the kalman filter. *University of North Carolina: Chapel Hill, North Carolina, US.*