



UNIVERSITÀ DI BRESCIA
FACOLTÀ DI INGEGNERIA
Dipartimento di Ingegneria dell'Informazione

Laboratorio di Robotica Avanzata **Advanced Robotics Laboratory**

Corso di Robotica
(Prof. Riccardo Cassinis)

Interazione TS-Vision Kawasaki

Elaborato di esame di:

Bellini Davide

Consegnato il:

10 dicembre 2013

Sommario

Il manipolatore industriale Kawasaki, non avendo alcun sensore per la visione, non è in grado di percepire alcun oggetto all'interno del suo spazio di lavoro, ciò non gli permette di recuperare gli oggetti a meno che non abbiano una posizione nota o non vi sia qualcuno che tramite il Teach Pendant lo addestri a raggiungere quella determinata posizione.

Il progetto realizzato ha permesso di mettere in comunicazione tra loro il manipolatore industriale Kawasaki e il software di visione TS-Vision in modo da automatizzare la capacità del manipolatore di rilevare la presenza di un oggetto all'interno del suo spazio di lavoro e di poter interagire con esso.

Per dimostrare questa nuova funzionalità è stato realizzato un programma di esempio che permetta, tramite una telecamera 2D e il software di visione TS-Vision, di individuare tre diversi oggetti disposti nel piano di lavoro del robot, di recuperarne le coordinate spaziali "x", "y" di comunicarle al Kawasaki, il quale, avrà il compito di recuperarle e riportarle in posizioni predefinite.

1. Introduzione

Prima di iniziare a parlare del progetto verranno mostrate e spiegate le diverse componenti, sia **software** che **hardware**, impegnate per la realizzazione del progetto.

1.1. Kawasaki

Il Kawasaki RS03N è un manipolatore industriale fisso a sei gradi di libertà ed è composto da tre parti fondamentali :

1. Braccio + End effector
2. Controller
3. Teach Pendant

1.1.1. Braccio ed End effector

Il braccio (Fig. 1 - Braccio) oltre a rappresentare la parte più caratteristica del robot, in quanto è la componente che si muove ed interagisce con il mondo, in base alla sua conformazione fornisce un certo grado di libertà al robot.

Per quanto riguarda il Kawasaki esso ha un grado di libertà pari a sei che sono anche il numero di giunti di cui è composto il braccio



Fig. 1 - Braccio

Nella parte finale del braccio vi è collocata una pinza (Fig. 2 – Pinza) che permette al manipolatore di poter afferrare diversi oggetti. Questa pinza è controllata tramite un'elettrovalvola che ne permette la chiusura e l'apertura.



Fig. 2 - Pinza

Tramite la pinza è possibile afferrare la ventosa (Fig.3 – Ventosa), che permette di afferrare diversi oggetti. Questa ventosa è controllata tramite un'elettrovalvola che permette di attivarla.

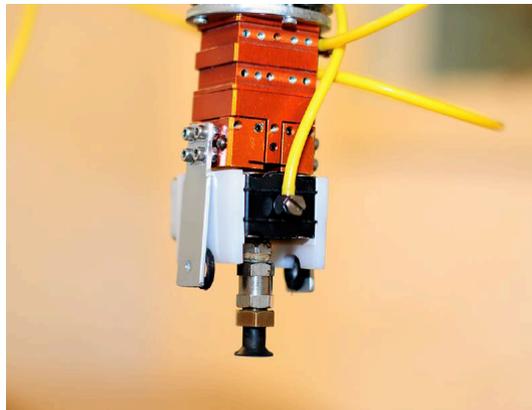


Fig. 3 - Ventosa

1.1.2. Controller

Il controller (Fig.4 - Controller) comprende:

1. Interruttore generale necessario per avviare il Robot
2. Il pannello Operativo dal quale è possibile scegliere tre diverse configurazioni in cui far partire il manipolatore :
 - a. 100% : modalità di verifica rapida del programma per la ripetizione automatica
 - b. TEACH : modalità che permette all'utente tramite l'utilizzo del Teach Pendant , che spiegheremo nel paragrafo successivo, di insegnare al robot come compiere determinate operazioni
 - c. REPEAT: modalità che permette di far eseguire in maniera automatica i programmi presenti

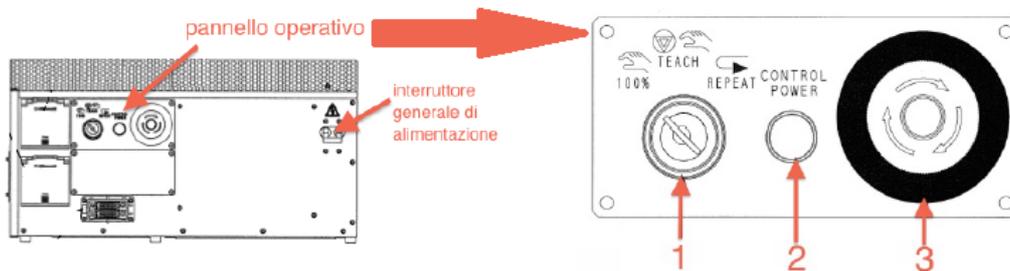


Fig. 4 - Controller

1.1.3. Teach Pendant

Il Teach Pendant (Fig. 5 – Teach Pendant) è un dispositivo che viene utilizzato dagli utenti per muovere il manipolatore in maniera manuale.

Questo dispositivo è composto da :

1. Uno schermo touchscreen
2. Tastiera Hardware
3. Pulsante per arrestare tutte le operazioni che sta compiendo il robot in caso di emergenza
4. Pulsante blocco insegnamento (T.LOCK)
5. Interruttore di sicurezza

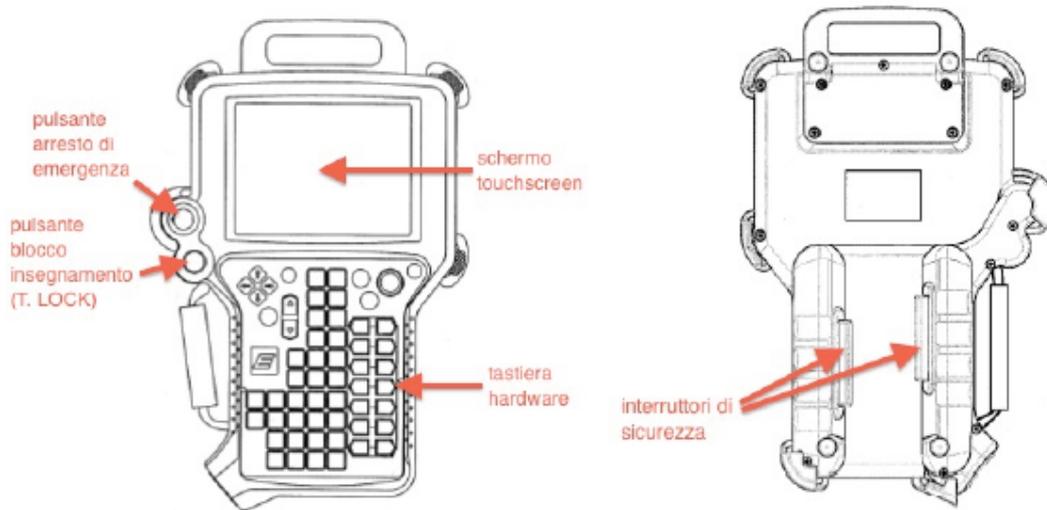


Fig. 5 - Teach Pendant

Per una trattazione più accurata riguardante il Kawasaki è possibile consultare il manuale operativo oppure una guida sintetica messa a disposizione da un altro studente [1][2]

1.2. Linguaggio di programmazione AS

Il linguaggio di programmazione usato per scrivere i programmi, che permettono al robot di eseguire determinate operazioni in modalità REPEAT, è "AS".

I comandi che si possono utilizzare sono suddivisi in due macro-sezioni :

1. Comandi monitor: programmi che possono essere eseguiti solo nella modalità monitor del sistema AS
2. Comandi Programma: comandi utilizzati per la programmazione del robot

Tramite questo linguaggio di programmazione è possibile programmare il robot per fargli compiere i diversi movimenti, inoltre, grazie a questo linguaggio di programmazione è possibile, tramite un'apposita comunicazione TCP/IP, mettere in comunicazione tra di loro il Kawasaki e il TS-Vision per lo scambio di informazioni.

Per ulteriori informazioni, per quanto riguarda i vincoli imposti dal linguaggio AS la modalità di programmazione e le operazioni che è possibili compiere, rimandiamo al manuale del Kawasaki e del linguaggio di programmazione presente sul portale ARL di robotica. [3]

1.3. TS-Vision e telecamera

Il sistema è stato concepito per eseguire la guida del robot mediante l'uso di telecamere in modo:

- affidabile
- veloce
- ripetitivo

Per soddisfare questi obiettivi, **Tiesse Robot Spa** ha strutturato il sistema di visione su una piattaforma standard, un PC con sistema operativo *WINDOWS*, ed ha sviluppato un apposito software "**TS-Vision**", che, attraverso una semplice interfaccia utente, utilizza appieno la potenzialità grafica di tali sistemi, per gestire l'esecuzione, la programmazione e l'interfaccia con l'operatore e con l'impianto con cui viene ad interagire (robot, PLC, altri programmi di windows, ecc.).

Il software TS-Vision che è presente in laboratorio è installato su computer (Fig. 6 – Computer) dove è installato il sistema “Windows 7”.



Fig. 6 - Computer

Per avviare il software è necessario collegare un'apposita chiavetta USB al calcolatore (Fig. 7 – Chiavetta USB con codici di autenticazione) contenente la licenza del TS-Vision.



Fig. 7 - Chiavetta USB con codici di autenticazione

All'avvio il software presenta una schermata (Fig. 8 – Schermata TS-Vision) contenente diversi menu:

- Progetto: necessario per creare un nuovo progetto e aprirne uno esistente
- Programma: necessario per creare un nuovo programma e aprirne uno esistente
- Strumenti: necessario per calibrare il dispositivo, gestire gli utenti, gestire i dispositivi collegati in rete e gestire le impostazioni del sistema

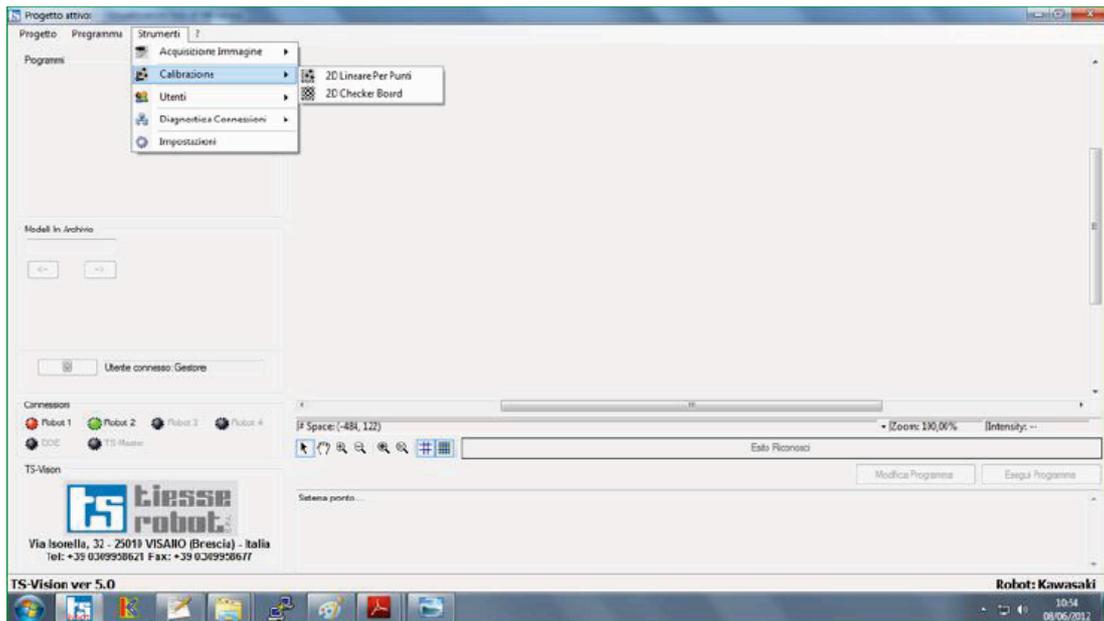


Fig. 8 - Schermata TS-Vision

Per permettere di acquisire le immagini al software è stato necessario l'installazione di una telecamera 2D posizionata sopra il Kawasaki (Fig. 9 – Telecamera).

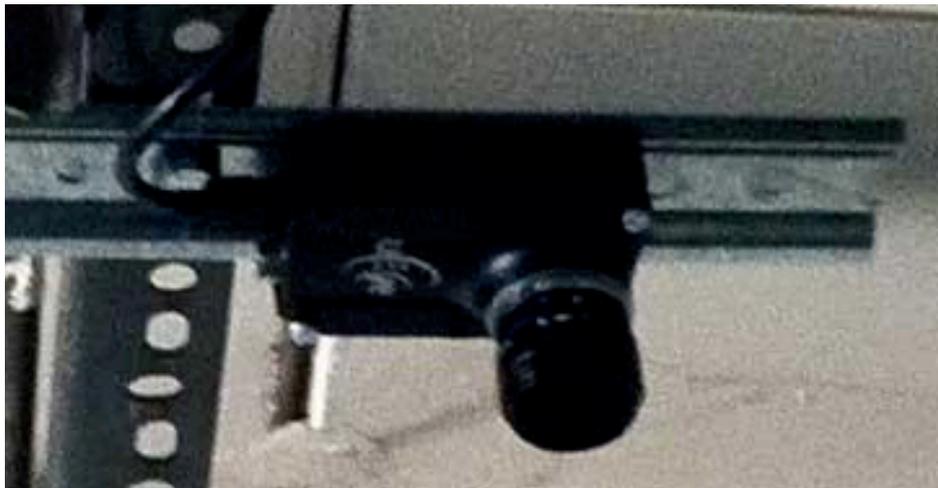


Fig. 9 - Telecamera

Per trattazione più dettagliata, che spiega in maniera esaustiva e nel dettaglio tutte le funzionalità messe a disposizione dal TS-Vision e tutte le operazioni necessarie per lavorare con questo software, rimandiamo alla relazione fatta da un altro studente e presente sul portale del corso di Robotica [4]

2. I problemi del Kawasaki

In questo capitolo verranno illustrati i problemi del Kawasaki, derivanti dal non aver nessun sensore che gli permetta di percepire gli oggetti presenti all'interno dello spazio di lavoro.

Questo è un problema in quanto, per far raggiungere al Kawasaki una determinata posizione è necessario tener presente che:

- Il manipolatore può raggiungere autonomamente posizioni note ovvero posizioni che l'utente ha già inserito precedentemente
- Il manipolatore durante il lavoro non è in grado di percepire gli oggetti presenti sulla sua traiettoria, quindi potrebbe incorrere in collisioni

Il fatto che il robot possa raggiungere solamente posizioni note, ovvero già predefinite in partenza, comporta che se venisse richiesto al robot di raggiungere una nuova posizione quest'ultimo non riuscirebbe a farlo autonomamente ma sarebbe richiesto l'intervento di un utente, tramite l'apposito dispositivo chiamato "teach pendant", che addestra il manipolatore a raggiungere la nuova posizione specificando la posizione che i giunti devono assumere per raggiungere l'oggetto.

Questo limita molto l'utilizzo del Kawasaki in quanto, se si dovesse richiedere, per esempio, al manipolatore di recuperare gli oggetti su un nastro trasportatore, non ne sarebbe capace in quanto la posizione di ogni oggetto sul nastro non è fissata a priori ma varia col tempo.

Un ulteriore problema è quello della collision avoidance, ovvero il manipolatore non è in grado, durante il suo movimento, di evitare la collisione con oggetti presenti sulla sua traiettoria. Questo è dovuto al fatto che non ha sensori per individuare eventuali ostacoli.

2.1. Il problema affrontato

Il problema affrontato, è il primo illustrato nel paragrafo precedente, quello relativo alla limitazione del manipolatore nel non riuscire a recuperare gli oggetti che non si trovano in posizioni note a priori.

Come già detto precedentemente per far raggiungere al manipolatore posizioni nuove è necessario che un utente, tramite l'apposito TEACH pendant, lo addestri correttamente; questo comporta i seguenti problemi :

1. Rallentamento della produttività del manipolatore: l'utente impiega un periodo di tempo maggiore rispetto a quello che sarebbe richiesto se invece il robot riuscisse a fare tutto in maniera autonoma.
2. L'imitazione per l'utilizzo: infatti come già detto il robot non riesce a compiere in maniera autonoma determinati compiti e questo limita l'utilizzo che se si può fare del manipolatore.

3. La soluzione adottata

In questo paragrafo vedremo la soluzione proposta per il problema visto precedentemente, la trattazione si divide in due parti:

- **spiegazione concettuale** in cui verrà analizzato il lavoro compiuto in maniera concettuale senza analizzare il codice scritto
- **programmazione Kawasaki TS-Vision** in cui verrà analizzato nel dettaglio il codice AS, utilizzato per la programmazione del robot, e la procedura eseguita per addestrare il TS-Vision.

3.1. Spiegazione Concettuale

Dopo aver analizzato il problema, che ha portato alla realizzazione di questo progetto, analizziamo la soluzione che si è progettata.

Per risolvere il problema della cecità del Kawasaki si è deciso di utilizzare TS-Vision, un apposito software di visione; questo, sfruttando una telecamera 2D, montata sopra il piano di lavoro del manipolatore, è in grado di rilevare la presenza di oggetti all'interno dello spazio di lavoro e di calcolarne la posizione.

Il TS-Vision, per riconoscere determinati oggetti e quindi restituirne la posizione, deve essere inizialmente istruito. La procedura da eseguire per compiere questo compito è la seguente: come prima cosa deve essere creato un progetto, una volta fatto deve essere inserito al suo interno un programma che contiene i modelli degli oggetti che verranno ricercati dal software.

Una volta fatto questo, tramite il protocollo TCP/IP, vengono messi in comunicazione tra di loro il TS-Vision e il Kawasaki; il primo addestra il robot a raggiungere il punto dove si trova un determinato oggetto all'interno dello spazio di lavoro, prendendo le veci dell'utente che manovra il robot tramite il "Teach Pendant", il secondo, una volta acquisite le coordinate, si reca sopra l'oggetto e compie una determinata azione.

In questo modo anche se la posizione degli oggetti dovesse cambiare continuamente il manipolatore riuscirebbe a lavorare in maniera del tutto autonoma senza richiedere la presenza dell'uomo.

Come prova che la soluzione adottata funziona, viene proposto un programma d'esempio per mostrare l'interazione tra il Kawasaki e il TS-Vision; vengono posti sul piano di lavoro del robot tre oggetti in posizioni del tutto casuali e si richiede che il manipolatore li recuperi e li posizioni ai lati del piano.

Vediamo più nel dettaglio:

1. vengono posizionati sul piano, in maniera del tutto casuale, tre forme (un cerchio un quadrato e un rombo)
2. il kawasaki richiede inizialmente al TS-Vision di fornirgli la posizione del cerchio, tramite una chiamata TCP/IP,
3. il software di visione fornisce i dati richiesti al robot
4. Ricevuti i dati il robot si reca nel punto indicato prende l'oggetto e spostandolo al lato del piano

questa procedura viene eseguita per tutte e tre le figure in maniera del tutto automatica senza che sia richiesta la presenza di un utente.

Questo come già detto prima permette di capire come adesso grazie a questa collaborazione il Robot sia riuscito a risolvere il suo problema e riesca a lavorare in maniera del tutto indipendente.

3.2. Programmazione Kawasaki TS-Vision

Nella prima parte di questo capitolo abbiamo focalizzato la nostra attenzione ad analizzare la situazione in maniera concettuale, ora invece passiamo ad analizzare tutto quello che concretamente è stato fatto per rendere operativo tutto il lavoro.

Qui di seguito verrà riportato il codice per:

- la comunicazione tra TS-Vision e Kawasaki.
- far muovere il Kawasaki.
- addestrare il TS-Vision a riconoscere le immagini poste all'interno dello spazio di lavoro del manipolatore.

3.2.1. Comunicazione tra TS-Vision e Kawasaki

La prima cosa che bisogna fare è quella di mettere in comunicazione tra di loro le due parti attivando una connessione TCP/IP.

In questo modo è possibile creare un canale di comunicazione che può essere utilizzato da entrambi per inviare e ricevere dati.

Come si può vedere dal codice riportato qui sotto (Fig.10 – Connessione) viene instaurata una connessione passando l'indirizzo IP del computer collegato in rete su cui è installato il programma TS-Vision e la porta su cui comunica.

```
.PROGRAM connettitsvisio()#0
  ipcam[1] = 192
  ipcam[2] = 0
  ipcam[3] = 2
  ipcam[4] = 97

  TCP_CONNECT socket_id,50001,ipcam[1]

  IF socket_id<0 THEN
    PRINT "Connessione fallita."
  ELSE
    PRINT "Connessione stabilita."
  END
.END
```

Fig. 10 - Connessione

Una volta instaurata la connessione è necessario creare una funzione che permetta al controller del Kawasaki di inviare una richiesta al software TS-Vision affinché questo possa cercare l'oggetto desiderato all'interno dello spazio di lavoro e restituirne le coordinate spaziali (Fig. 11 – Cerca Immagini).

```

PROGRAM searchimage(.ret,.$data)#0
$send_buf[1] = .$data
buf_n = 1
.ret = 1
TCP_SEND sret,socket_id,$send_buf[1],buf_n
IF sret<0 THEN
  .ret = -1
  PRINT "[searchImage] Errore di invio al server."
ELSE
  PRINT "[searchImage] Invio di ",.$data," corretto."
END
.END

```

Fig. 11 - Cerca Immagini

A questo punto il software di visione esegue il suo compito: cerca all'interno dello spazio di lavoro l'immagine richiesta, ne recupera le coordinate "x" "y", ne crea un pacchetto, le mette sul buffer e le invia al manipolatore; il Controller riceve il pacchetto, come mostrato sotto (Fig.12 – Ricevi dato TS-Vision), recuperando le diverse coordinate.

```

PROGRAM ricevitsvision()#0
.num = 0
TCP_RECV rret,socket_id,$recv_buff[1],.num
IF rret<0 THEN
  PRINT "[riceviTSVISION] Errore in ricezione: ",rret
  $recv_buff[1] = "000"
ELSE
  IF .num>0 THEN
    PRINT "[riceviTSVISION] Dati ricevuti: ",$recv_buff[1]
  ELSE
    $recv_buff[1] = "000"
  END
END
.END

```

Fig. 12 - Ricevi dato TS-Vision

3.2.2. Far muovere il Kawasaki

Come prima cosa, prima ancora di far comunicare tra di loro il manipolatore e il software di visione è necessario che il primo recuperi la ventosa (Fig. 13 – Prendi ventosa), posizionata alla sua destra, in modo da avere tutto il necessario per eseguire correttamente il suo compito.

```

PROGRAM bprendiogetto()#0
CALL prendi_ventosa
PRINT "inizio - TRANS(064 013 350 E"

```

Fig. 13 - Prendi ventosa

Una volta attivata la comunicazione tra il TS-Vision e il Kawasaki e acquisite le informazioni dal software di visione, quello che si ha a disposizione è una stringa contenente tutte le coordinate relative alla posizione dell'immagine che si vuole recuperare; è necessario interpretarla (Fig.14 – Coordinate X Y), tramite delle apposite funzioni forniteci dal linguaggio AS, in modo da poter usare le coordinate "x" e "y" e poter far muovere il robot nelle posizione in cui si trova la figura che si desidera spostare, una

volta raggiunta tramite il comando TRANS, che permette di dare al manipolatore la posizione da raggiungere nel formato corretto, tramite il comando JMOVE si porta il robot nella posizione desiderata.

```

PRINT "Questi sono i dati forniti dal TS Vision = " + $recv_buff[1]
.$rx0 = $recv_buff[1]
$temp = $DECODE(.$rx0,"",0)
$temp = $DECODE(.$rx0,"",1)
$temp = $DECODE(.$rx0,"",0)
$temp = $DECODE(.$rx0,"",1)
$temp = $DECODE(.$rx0,"",0)
x0 = VAL($temp)
TYPE "x = ",x0
$temp = $DECODE(.$rx0,"",1)
$temp = $DECODE(.$rx0,"",0)
y0 = VAL($temp)
TYPE "y = ",y0
$temp = $DECODE(.$rx0,"",1)
$temp = $DECODE(.$rx0,"",0)
AO = VAL($temp)

$temp = $DECODE(.$rx0,"",1)
POINT posa = TRANS(-x0+10,-y0+14,0,0,0)

```

Fig. 14 - Coordinate X Y

Successivamente si fa recuperare l'oggetto al robot tramite la ventosa e lo si posiziona al lato del piano di lavoro .

Analizzando la situazione risulterà subito chiaro che il Kawasaki ha l'origine dei propri assi cartesiani in una posizione diversa rispetto a quella del TS-Vision e che quindi per far sì che i due comunichino correttamente è necessario che i due centri degli assi coincidano; per far ciò viene usata la funzione "BASE"(Fig.15 – Base Nuova Origine) in AS che permette di spostare l'origine degli assi del manipolatore facendola combaciare con quella del sistema di visione.

```

PROGRAM bprendiogetto)#0
CALL prendi_ventosa
POINT inizio = TRANS(264.813,350.514,-13.566,-169.802,178.883,45.284,611)
JMOVE inizio
POINT nuovaorigine = TRANS(106.495,440.297,-13.566,-169.802,178.883,45.284,611)
BASE nuovaorigine

```

Fig. 15 - Base Nuova Origine

Tutto quello appena descritto viene ripetuto anche per le altre due immagini come viene riportato nel codice qui sottostante (Fig. 16 – Interazione Kawasaki TS-Vision per cerchio) (Fig. 16 – Interazione Kawasaki TS-Vision per rombo)

```

POINT inizio = TRANS(264.813,350.514,-13.566,-169.802,178.883,45.284,611)
JMOVE inizio
POINT nuovaorigine = TRANS(106.495,440.297,-13.566,-169.802,178.883,45.284,611)
BASE nuovaorigine
;

.$data = "M"+$ENCODE(/I1,1)+"@"
CALL searchimage(.ret,.$data)
CALL ricevitsvision
PRINT "Questi sono i dati forniti dal TS Vision = " + $recv_buff[1]
.$rx0 = $recv_buff[1]
$temp = $DECODE(.$rx0,"",0)
  $temp = $DECODE(.$rx0,"",1)
  $temp = $DECODE(.$rx0,"",0)
  $temp = $DECODE(.$rx0,"",1)
  $temp = $DECODE(.$rx0,"",0)
  x0 = VAL($temp)

  $temp = $DECODE(.$rx0,"",1)
  $temp = $DECODE(.$rx0,"",0)
  y0 = VAL($temp)

  $temp = $DECODE(.$rx0,"",1)
  $temp = $DECODE(.$rx0,"",0)
  AO = VAL($temp)

  $temp = $DECODE(.$rx0,"",1)
  POINT posa = TRANS(-x0+10,-y0-14,0,0,0,0)
  LAPPRO posa,84.37

```

Fig. 16 - Interazione Kawasaki TS-Vision per cerchio

```

POINT inizio = TRANS(264.813,350.514,-14.087,-169.802,178.883,45.284,611)
JMOVE inizio
POINT nuovaorigine = TRANS(106.495,440.297,-13.566,-169.802,178.883,45.284,611)
BASE nuovaorigine
inserire movimento del robot;
.$data = "M"+$ENCODE(/I1,2)+"@"
CALL searchimage(.ret,.$data)
CALL ricevitsvision
PRINT "Questi sono i dati forniti dal TS Vision = " + $recv_buff[1]
.$rx0 = $recv_buff[1]
$temp = $DECODE(.$rx0,"",0)
  $temp = $DECODE(.$rx0,"",1)
  $temp = $DECODE(.$rx0,"",0)
  $temp = $DECODE(.$rx0,"",1)
  $temp = $DECODE(.$rx0,"",0)
  x0 = VAL($temp)-30
  $temp = $DECODE(.$rx0,"",1)
  $temp = $DECODE(.$rx0,"",0)
  y0 = VAL($temp)

  $temp = $DECODE(.$rx0,"",1)
  $temp = $DECODE(.$rx0,"",0)
  AO = VAL($temp)

  $temp = $DECODE(.$rx0,"",1)
  POINT posa = TRANS(-x0-15,-y0-20,0,0,0,0)
  JMOVE posa
  LAPPRO posa,84.37

```

Fig. 17 - Interazione Kawasaki TS-Vision per quadrato

3.2.3. Addestrare il TS-Vision a riconoscere le immagini poste all'interno dello spazio di lavoro del manipolatore

In questo paragrafo vengono riportate tutte le operazioni fatte per addestrare il software di visione a riconoscere correttamente tutte le immagini all'interno dello spazio di lavoro.

La prima cosa da fare è quella di creare il “PROGETTO” in cui verrà inserito il programma per il riconoscimento delle immagini.

Questa operazione è possibile farla tramite l'apposita voce del menu: “Progetto→Nuovo”

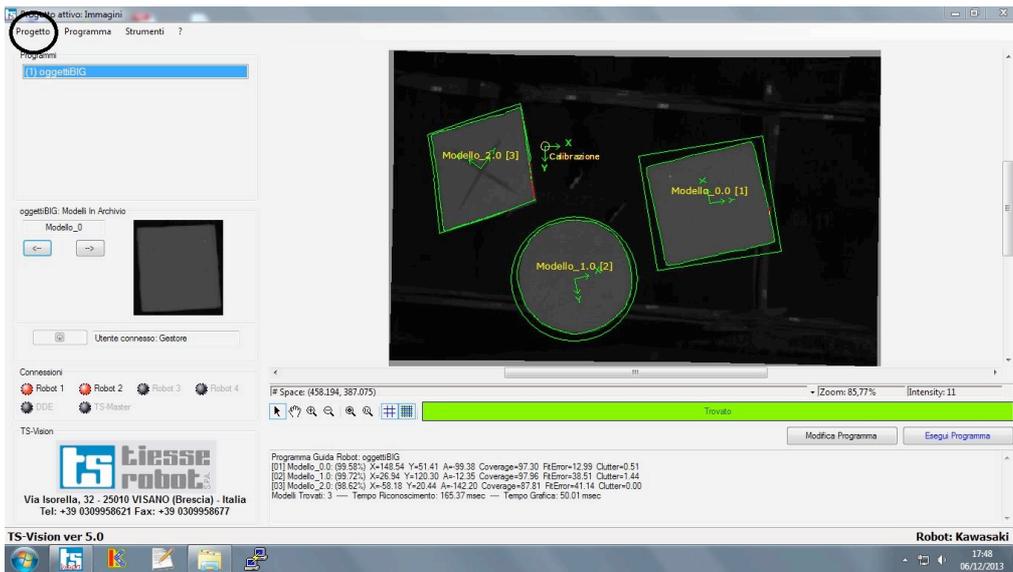


Fig. 18 - Progetto

La seconda cosa da fare, una volta creato il progetto, è creare un “PROGRAMMA” che conterrà tutti i modelli delle immagini che dovranno essere ricercate.

Questa operazione è possibile farla tramite l'apposita voce del menu: “Programma→Nuovo”

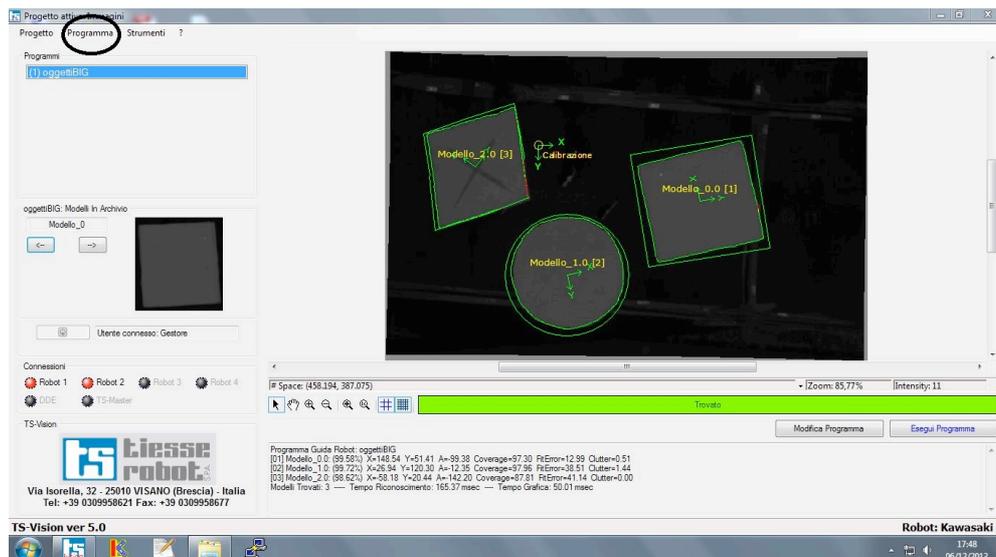


Fig. 19 - Progetto

L'ultima operazione richiesta per il corretto addestramento del dispositivo è quella di creare gli appositi "MODELLI", per questo è richiesto che il dispositivo acquisisca le immagini che raffigurano gli oggetti che dovrà cercare. Per acquisire questi modelli basta andare sulla voce di menu "Strumenti → Acquisizione immagine" in questa sezione del programma sarà possibile acquisire l'immagine tramite la telecamera 2D e trasformare questa immagine in un modello da inserire all'interno del progetto.

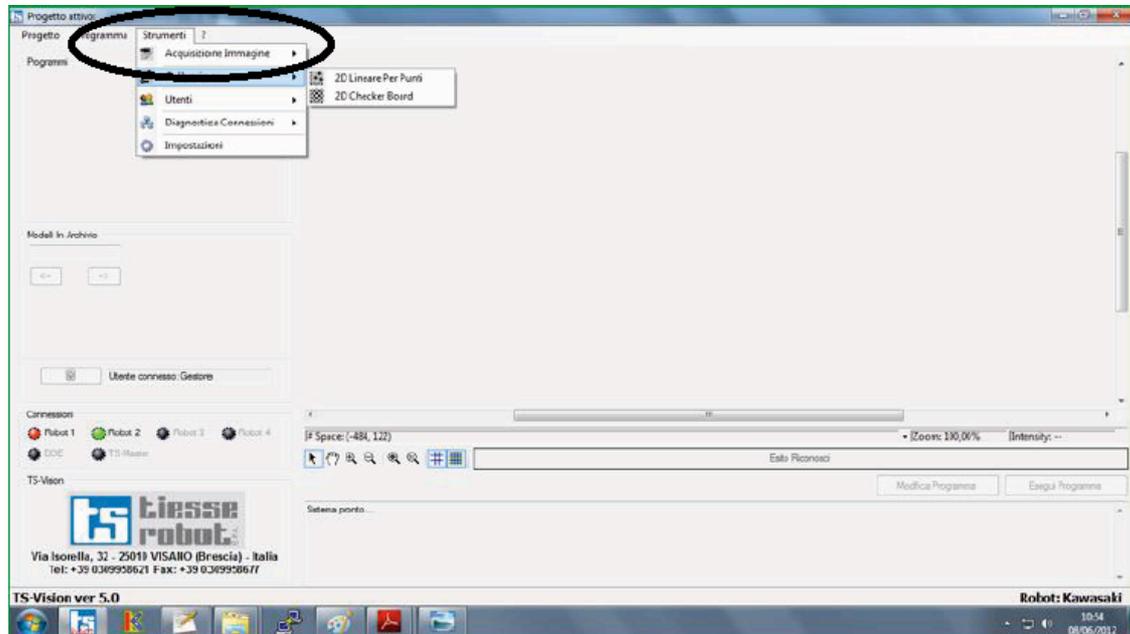


Fig. 20 - Modello

La figura qui di seguito (Fig.21 – TS-Vision) mostra come il software si presenta una volta che viene eseguito il programma e vengono rilevate tutte e tre le immagini. Nella parte bassa si può vedere come il software mostri a video tutti i dati che vengono inviati al Robot Kawasaki, ovvero le coordinate spaziali "X" e "Y" il numero del modello.

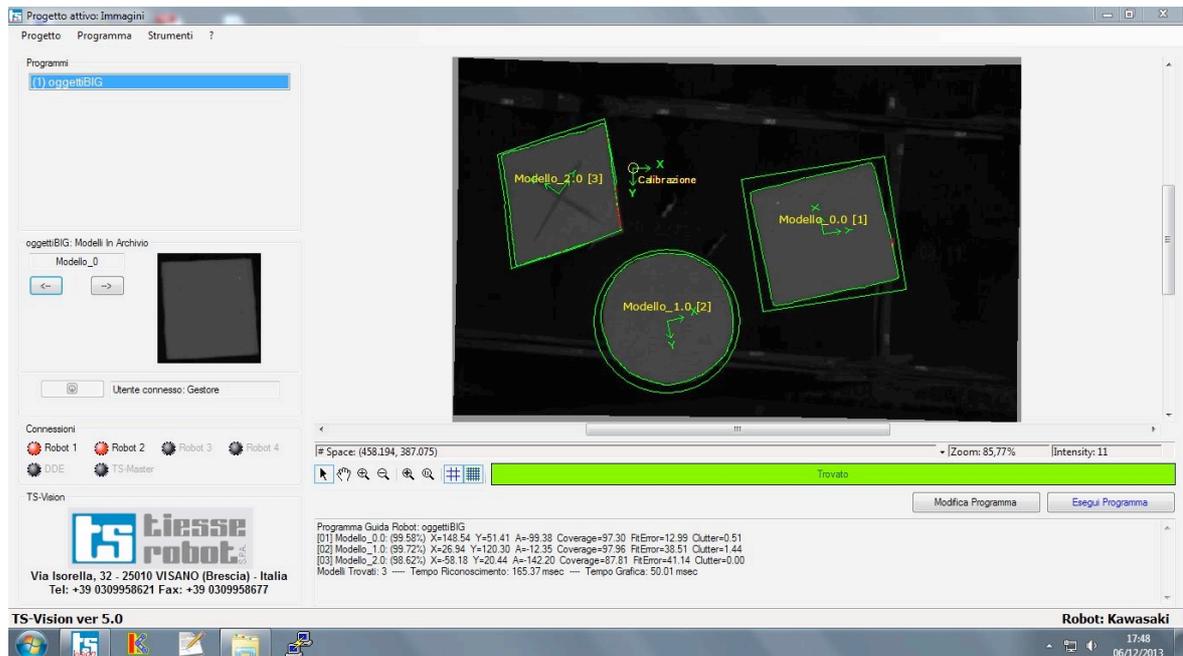


Fig. 21 - TS-Vision

Per una trattazione più accurata, su tutti i passi che devono essere fatti per riuscire ad addestrare correttamente il TS-Vision, rimandiamo alla documentazione presente sul portale del corso [4]

4. Modalità operative

Una volta vista la soluzione proposta vengono mostrati tutti i componenti necessari al corretto funzionamento del programma.

4.1. Componenti necessari

Le componenti principali per il corretto funzionamento del mio programma sono le seguenti :

- **Manipolatore Kawasaki**



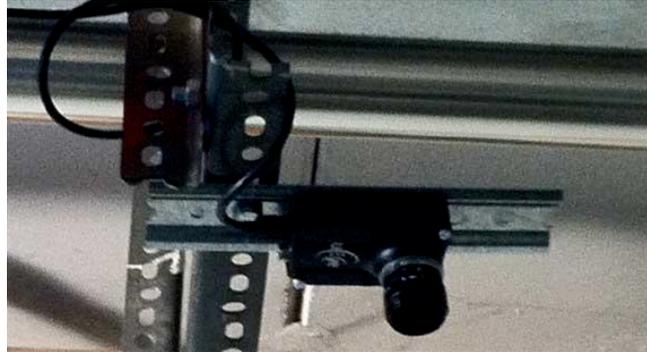
- **Teach Pendant** necessario per avviare il programma scritto in AS installato sul controller del Kawasaki con il nome di “**bprendioggetto**”



- **Il computer** presente in laboratorio con installato sopra il software TS-Vision il quale deve essere collegato in rete tramite Ethernet per poter comunicare con il Controller del manipolatore



- **La telecamera 2D** installata sopra il manipolatore, che deve essere collegata al PC in modo da essere utilizzata dal software di visione



- **basetta nera**: Questo si è reso necessario perché la telecamera è in bianco e nero e quindi aumentando il contrasto con le figure, di colore bianco, si rende più facile per il software l'identificazione dell'immagine all'interno dello spazio di lavoro.

Durante la fase di sviluppo si utilizzava una basetta in cartone sostituita in seguito da un'altra fatta di compensato sempre di colore nero (Fig.22 – Basetta) . Questo è stato fatto perché grazie alle proprietà del compensato è possibile avere a disposizione una piattaforma di spessore ridotto, poco ingombrante, di poco peso e rigida. Quest'ultima caratteristica è molto importante perché si è notato, durante la fase di testing, che nella fase di presa dell'oggetto la basetta in cartone si fletteva e si spostava causando alcune volte la mancata presa dell'oggetto da parte del robot.



Fig. 22 - Basetta

- **Tre oggetti** rappresentanti le forme: “quadrato”, “cerchio”, “rombo”

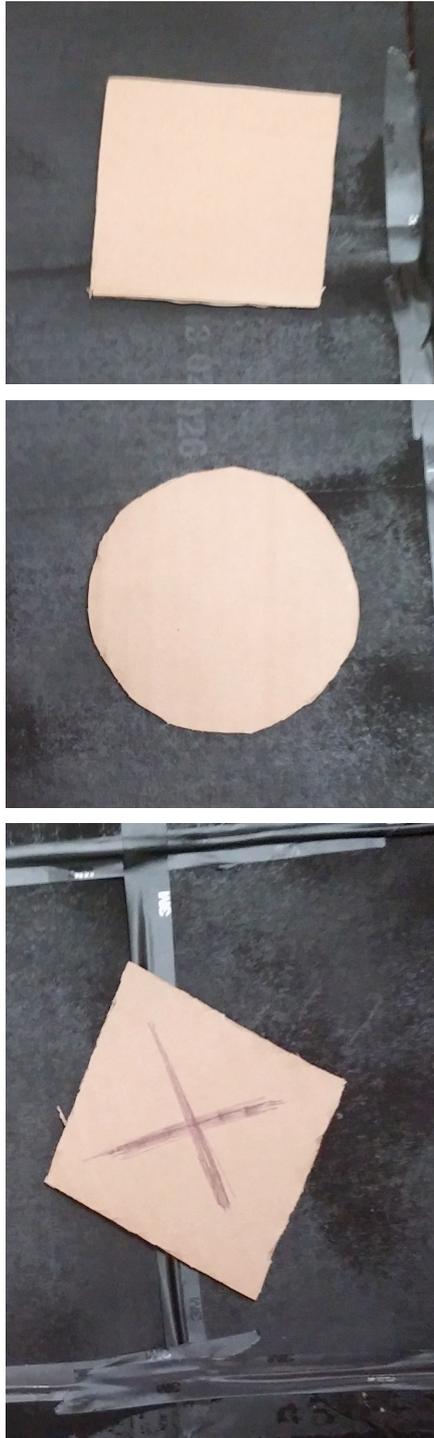


Fig. 23 - Figure

Questi tre oggetti sono stati realizzati in cartone e tutte e tre della stessa altezza in quanto la telecamera utilizzata dal TS-Vision essendo in 2D non permette di acquisire le informazioni relative all'asse "Z" e quindi si è dovuto adottare un valore standard per sopperire a questa mancanza.

5. Conclusioni e sviluppi futuri

Dopo aver fatto una visione approfondita di tutto quello che è stato fatto possiamo concludere dicendo che, grazie all'interazione tra il software di visione TS-Vision, il robot Kawasaki ha risolto il suo problema riguardante il fatto di non riuscire ad identificare oggetti nella sua sezione di lavoro.

Gli sviluppi futuri da apportare al programma posso essere:

- l'utilizzo di un numero maggiore di telecamere per riuscire ad acquisire più informazioni oltre alle sole coordinate "x" e "y" come ad esempio anche la coordinata "z" che mi fornisce l'altezza dell'oggetto e quindi mi permette di non lavorare con oggetti della stessa altezza, come è stato fatto, ma con altezze diverse.
- Una telecamera con una maggior risoluzione per permettere al software di poter utilizzare al meglio tutto la sua potenzialità e poter riconoscere oggetti non solo per la forma ma anche per il colore.

Bibliografia

- [1] Kawasaki Heavy Industries : “Operative Manual”
- [2] Stefano Saccani : “Guida all’utilizzo del robot Kawasaki RS03N”
- [3] Daniele Petre : “Guida introduttiva all’utilizzo del robot Kawasaki RS03N”
- [4] Luca Pedretti Erick Tanghetti : “Primer sul sistema TS-Vision”

Indice

SOMMARIO	1
1. INTRODUZIONE	1
1.1. Kawasaki	1
1.2. Linguaggio di programmazione AS	4
1.3. TS-Vision e telecamera	4
2. I PROBLEMI DEL KAWASAKI.....	7
2.1. Il problema affrontato	7
3. LA SOLUZIONE ADOTTATA	8
3.1. Spiegazione Concettuale	8
3.2. Programmazione Kawasaki TS-Vision	9
4. MODALITÀ OPERATIVE	16
4.1. Componenti necessari	16
5. CONCLUSIONI E SVILUPPI FUTURI.....	19
BIBLIOGRAFIA	20
INDICE	21