GERAÇÃO DE PROGRAMAS PARA ROBÔS INDUSTRIAIS EM AMBIENTE CAD

INDUSTRIAL ROBOT PROGRAM GENERATION VIA CAD SYSTEM

Carlos Norberto VETORAZZI JÚNIOR*
Prof. Dr. Geraldo Nonato TELLES**

ABSTRACT

The work described there is the development of a system to automatically generate robot programs for PCB (Printed Circuit Board) components insertion tasks, with the input of CAD files describing the lay-out of the robotic cell, and some files with information about the components of assembly task. The system plays the hole of finding the cordinates to be programmed - wich takes a lot of improductive time if manually done - and check for interference in the robotic insertion. This can lead to a more flexible way to programming a robot.

KEY WORDS: Robots, programs, automation, CAD - Computer Aided Design.

RESUMO

É descrito o desenvolvimento de sistema que gera automaticamente programas para inserção de componentes eletrônicos em placas de circuito impresso, a partir de arquivo CAD que descreve o "lay-out" da célula de fabricação, além de arquivos com informações sobre os componentes e as tarefas de montagem. O sistema determina as coordenadas a serem programadas - o que demandaria um tempo improdutivo grande se feito manualmente - e verifica a ocorrência de interferência na tarefa de inserção. O sistema desenvolvido possibilita uma flexibilização na programação de robô.

PALAVRAS-CHAVE: Robôs, programas, automação, CAD - Computador Auxiliando o Desenho.

INTRODUÇÃO

A montagem manual de componentes em cartões de circuito impresso (PCBs) é extremamente tediosa, sendo que atualmente utiliza-se com vantagens robôs industriais para essas tarefas (du Feu 1986),(Supinski 1991). Os robôs empregados nessas atividades são normalmente previamente programados através de linguagens textuais, e nestes programas encontram-se os valores das coordenadas, dadas no espaço de trabalho cartesiano do robô.

O sistema aqui apresentado utiliza como plataforma um microcomputador padrão PC, utilizando o sistema CAD AutoCAD R10 da Autodesk como fonte de dados. Os testes de validação foram feitos em um robô IBM 7535 do tipo SCARA (***), com linguagem AML para programação do robô (****).

A motivação para o desenvolvimento desse sistema reside na dificuldade de se determinarem corretamente as coordenadas que deverão constar no programa em AML. Essa obtenção é feita normalmente utilizando-se o robô como digitalizador dessas coordenadas, o que além de levar um tempo considerável, utiliza o robô de maneira improdutiva. Mas essa determinação poderia ser feita a partir de informações previamente disponíveis em arquivos CAD. Esses arquivos seriam os desenhos dos componentes, do alimentador, do dispositivo de fixação, das placas e do "lay-out" da célula, onde estariam as relações entre todos os elementos envolvidos. Se esses desenhos estão em uma escala adequada, e um desenho CAD não é mais do que um plano cartesiano onde cada

(*)Carlos Norberto Vetorrazi Jr. - Doutorando em Eng. Mecânica - UNICAMP.
(**) Geraldo Nonato Telles - Professor convidado do programa de Mestrado no convênio PUCCAMP/UNICAMP.
(***) SCARA: Selective Compliance ARticulated Arm
(****) AML: A Manufacturing Language

elemento do desenho pode ser descrito em termos de coordenadas desse plano cartesiano, então podemos "re-irar" as coordenadas para o programa desses desenhos.

Nesse projeto existe uma simplificação que consis-e no enfoque bidimensional (2D) do problema, tendo em vista que o robô em questão não tem servo-controle sobre o eixo Z; os parâmetros para este eixo podem ser apenas alto ("UP") e baixo ("DOWN"), tendo-se que obrigatoriamente projetar os alimentadores e dispositi-vos de fixação das placas de maneira que os componen-es são carregados e montados em uma mesma cota Z (mecanicamente determinada).

Esse projeto também não prevê o controle para se evitarem colisões, havendo necessidade de eventuais depurações (manuais) do programa após a simulação gráfica do mesmo, inserindo-se pontos intermediários (via-points), de forma a se evitar colisões da garra do robô com os outros dispositivos.

A utilização desse sistema é feita "off-line", indo de encontro a modernas tendências (inclusive Engenharia Simultânea), onde as atividades relativas à manufatura são associadas e concomitantes com as atividades de projeto, compartilhando uma mesma base de dados, havendo uma integração CAD/CAM, cujo objetivo final é o CIM(Jacobs 1991).

A automatização da montagem de componentes eletrônicos através de equipamentos dedicados (automação rígida) só é justificada para grandes volu-mes, tendo em vista as dificuldades de preparação dos sistemas ("set-up"). Assim, onde temos uma produção de menos de 100.000 PCBs por ano, ou mais de 100 tipos diferentes de PCBs(Supinski 1991), temos necessidade de flexibilidade, justificando o uso de sistemas robotizados programáveis, sendo importante a geração rápida e precisa de programas.

O emprego de Robôs não está livre de problemas, entretanto. Como vimos, existe a necessidade de uma integração CAD/CAM, de maneira a racionalizar a gera-ção dos programas para a manufatura.

OS ARQUIVOS CAD

É possível extrair as coordenadas de arquivos CAD. No sistema CAD utilizado é possível operar-se com "blocos" ("AutoCAD Reference Manual"). Cada bloco é um conjunto de entidades gráficas (retas, círculos), asso-ciadas como uma única entidade (o bloco). Assim, o desenho de um componente pode ser tratado como um bloco, bem como o desenho de uma placa. O comando ATTEXT (ATTribute EXTraction) do AutoCAD gera uma lista dos blocos utilizados em um desenho, com as respectivas coordenadas. Então, ao tratarmos os elementos (componentes, placas) como blocos em um desenho de "lay-out", podemos determinar as coordenadas de cada bloco, e estas coordenadas podem ser associadas com as coordenadas reais dos elementos na tarefa de montagem (Vetorazzi Jr., e Telles G.N. 1994).

Isto é feito construindo-se dois arquivos CAD (desenhos) de "lay-out" da montagem. Esse "lay-out" corresponde a uma representação do espaço de trabalho do robô. Então os desenhos dos elementos envolvidos no processo de montagem são posicionados (como blocos) nesse "lay-out", de maneira a representar a situação real. Um "lay-out" seria equivalente ao estado inicial do siste-ma, desenhos do alimentador e dos componentes. O outro "lay-out" seria equivalente ao estado final do siste-ma, quando os componentes encontram-se montados nas placas; então a montagem é feita posicionando-se o dispositivo de fixação no espaço de trabalho, as placas no dispositivo de fixação e os componentes nas placas (Vetorazzi Jr.,C. N. e Telles G. N. 1994) e (Vetorazzi Jr., C. N.1994)(figura 1).

Então utilizamos o comando ATTEXT para cada arquivo de "lay-out" e temos as coordenadas de cada bloco utilizado em cada "lay-out". As coordenadas que são importantes são as que correspondem aos componentes que não estão no alimentador e quando estão montados nas placas. Essas coordenadas correspondem àquelas que temos que colocar no programa do robô e que infor-mam a este onde deve ir buscar os componentes e para onde deve levá-los. Em princípio, as outras coordenadas não interessam. Os desenhos do alimentador, do dispositivo de fixação e das placas, são usados como auxiliares para o correto posicionamento dos componentes, já que na situação real eles realmente estarão no alimentador e serão colocados nas placas, mas o que realmente interessa são as coordenadas dos componentes. A figura 2 mostra um arquivo típico gerado pelo comando ATTEXT, e a figura 3 mostra o desenho correspondente.

Essas coordenadas correspondem aos sistemas de referência utilizados para desenhar cada bloco em rela-ção ao sistema de referência do "lay-out", que deve corresponder ao do próprio robô. Conseqüentemente, esses sistemas de referência devem ser criteriosamente escolhidos, sendo interessante que correspondam a al-gum atributo físico do elemento que será usado na montagem (Vetorazzi Jr.,C.N. 1994) (por exemplo um pino no componente e um furo no cartão) (figura 4). A seguir é necessário "corrigir" essas coordenadas em função da localização do sistema de referência, do tamanho do componente, do tamanho e da orientação da garra e da orientação do componente.

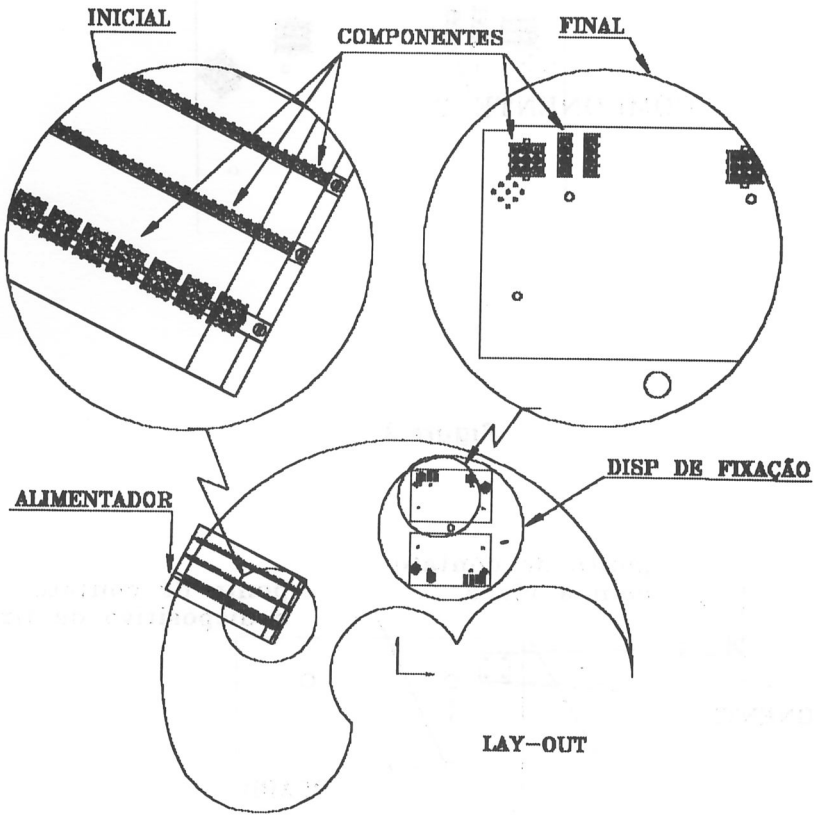

Figura 1

Nivel de aninham.	Nome do Bloco	Pos x	Pos y	Pos z	Num de ordem	Angulo r	Nome do componente
1,	'COMP1',	-1.96,	79.51,	0.00,	1,	0.00,	'componente1'
1,	'COMP1',	135.12,	76.86,	0.00,	2,	0.00,	'componente1'
1,	'COMP1',	175.65,	59.74,	0.00,	3,	315.00,	'componente1'
1,	'COMP1',	-15.96,	65.51,	0.00,	4,	315.00,	'componente1'
1,	'COMP2',	27.94,	79.51,	0.00,	5,	0.00,	'componente2'
1,	'COMP2',	44.34,	79.51,	0.00,	6,	0.00,	'componente2'

Figura 2

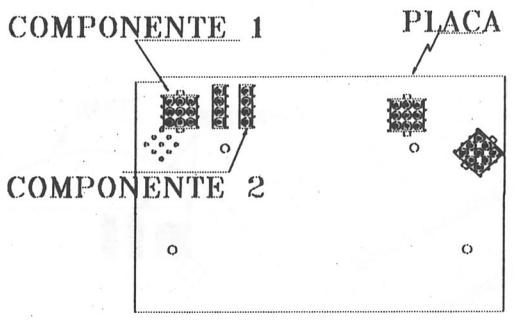

Figura 3

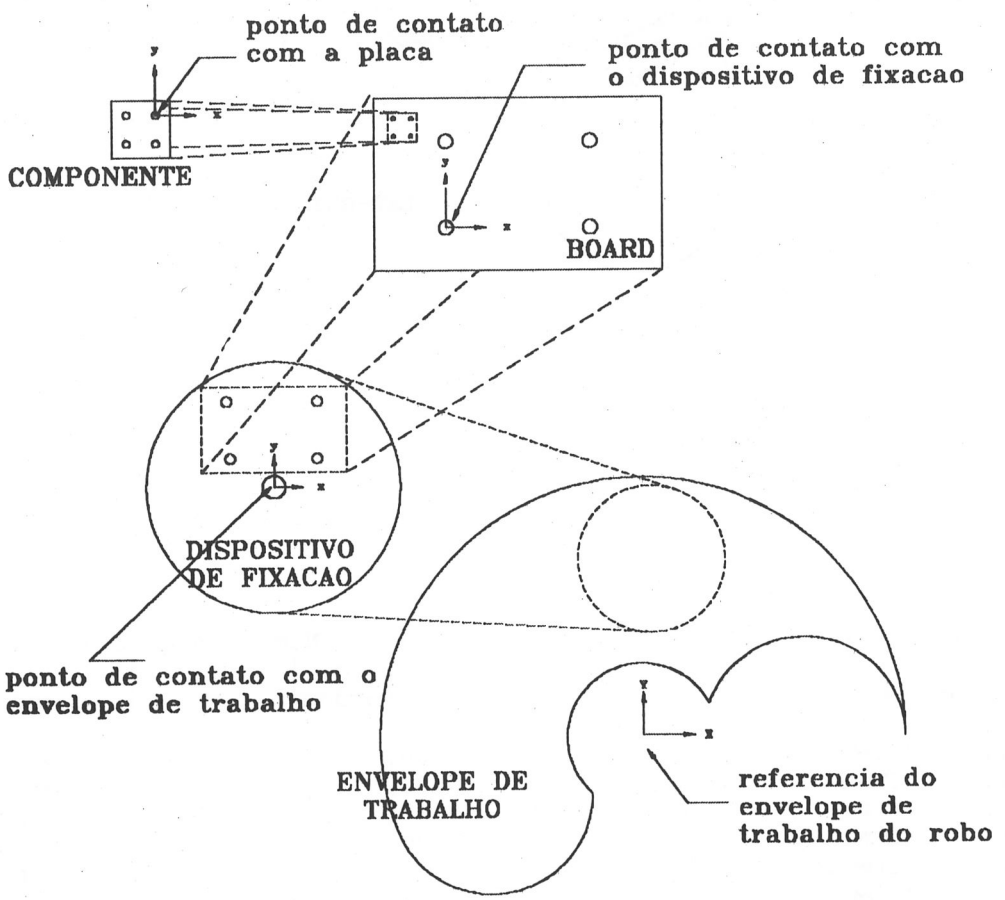

Figura 4

Então ajusta-se essas coordenadas de maneira que correspondam à coordenada da garra do robô. Com estas informações pode-se então determinar se vai haver interferência entre a garra e componentes previamente montados, o que implica na definição de algumas limitações e se necessário, na determinação da seqüência de inserção dos componentes, ou mesmo na indicação de componentes a serem inseridos manualmente. Atualmente essas informações são fornecidas manualmente pelo usuário, por ocasião do desenho dos elementos.

Outra atribuição do sistema (Vetorazzi Jr., C.N. 1994) é a definição de pontos intermediários, para a otimização do ciclo de montagem. Próximo a cada ponto-alvo, é definido um ponto intermediário, e entre estes, o robô se movimenta com alta acurácia (o que implica baixa velocidade), e entre os pontos intermediários o robô se movimenta com alta velocidade (o que implica baixa acurácia). Uma maior acurácia nas proximidades dos pontos-alvo reduz a possibilidade de erros do robô para atingir esses pontos, e uma maior velocidade entre os pontos intermediários faz com que o tempo de ciclo seja menor.

O SISTEMA

As coordenadas estão prontas para serem incorporadas ao programa. O programa é um conjunto de subrotinas específicas (uma subrotina para pegar o componente, outra para inserir o componente, etc). As subrotinas são as mesmas para várias montagens diferentes, somente as coordenadas mudam, então temos um problema do tipo "complete a sentença", onde os espaços em branco correspondem às coordenadas. E as sentenças com os espaços seriam comandos da linguagem, formando um "esqueleto" a ser preenchido.

Na verdade, este "esqueleto" do programa é interno ao sistema, e não um elemento a parte. Isto não é uma boa prática, porque é interessante escreverem-se pré-processadores para outras linguagens, de forma a se gerar um programa genérico (internamente ao sistema), que contenha a lógica do processo, e então seria possível gerar um programa em uma linguagem específica, traduzindo o programa genérico para esta linguagem, mediante um pré-processamento.

O sistema tem como base um menu para o direcionamento do trabalho, onde cada opção aciona um módulo específico. São três módulos principais; no primeiro escolhem-se os arquivos de "lay-out", e é feita uma preparação dos dados (agrupamento, ordenação); no segundo módulo são feitos todos os cálculos de coordenadas, verificação de interfaces, determinação de seqüência de inserção e determinação das coordenadas "corrigidas"; no terceiro módulo é feita a montagem do programa do robô, gerando os códigos do programa e incluindo as coordenadas.

As figuras 5 e 6 mostram dois programas gerados pelo sistema. A figura 5 corresponde a uma situação onde não há restrições devido a interferências, possibilitando o uso da primitiva "ITERATE" da linguagem AML/E, agrupando os elementos do mesmo tipo e tornando os programas mais concisos.

A figura 6 corresponde a uma situação onde há restrições.

```
- DEFINICOES DO ARQUIVO DEFAULT.AML :

PRECISAO : NEW 1;
VELOCIDADE : NEW 10;
ESPERA : NEW 1.5;
HOME : NEW PT( 650, 0, 0 );

-- COORDENADAS DOS ALIMENTADORES

S1  : NEW PT(  -93.99, 358.23,-135.22); -- componente1
S2  : NEW PT( -125.31, 410.39,-135.22); -- componente2

-- JA EXISTE UM ALIMENENTADOR PARA O TIPO ABAIXO
-- S2  : NEW PT( -163.64, 448.43,-135.22); -- componente2

-- COORDENADAS DOS COMPONENTES NA PLACA

T1_3  : NEW PT(  23.29, 405.82, -45.00); -- componente1
T1_4  : NEW PT(  122.10, 500.88, -45.00); -- componente1
T1_5  : NEW PT(  169.86, 514.53,  0.00); -- componente1
T1_9  : NEW PT(  341.21, 325.18, 135.00); -- componente1
T1_10 : NEW PT(  242.40, 230.12, 135.00); -- componente1
T1_11 : NEW PT(  194.64, 216.47,-180.00); -- componente1
T2_7  : NEW PT(  56.03, 440.68, -45.00); -- componente2
T2_13 : NEW PT(  308.47, 290.32, 135.00); -- componente2

-- PONTOS INTERMEDIARIOS PARA ALIMENTADOR E
PLACAS

WT1_3  : NEW PT(  14.10, 401.87, -45.00); -- componente1
WT1_4  : NEW PT(  114.20, 494.75, -45.00); -- componente1
WT1_5  : NEW PT(  161.56, 508.95,  0.00); -- componente1
WT1_9  : NEW PT(  331.24, 325.94, 135.00); -- componente1
WT1_10 : NEW PT(  233.12, 233.84, 135.00); -- componente1
WT1_11 : NEW PT(  185.82, 221.19,-180.00); -- componente1
WS1  : NEW PT(  -85.17, 353.52,-135.22); -- componente1
WT2_7  : NEW PT(  46.17, 439.02, -45.00); -- componente2
WT2_13 : NEW PT(  298.85, 293.05, 135.00); -- componente2
WS2  : NEW PT( -115.69, 407.66,-135.22); -- componente2

-- JA EXISTE UM ALIMENENTADOR PARA O TIPO ABAIXO
-- WS2  : NEW PT( -115.69, 407.66,-135.22); -- componente2
```

```
T1 : NEW < T1_3, T1_4, T1_5, T1_9, T1_10, T1_11 >;

WT1  :  NEW  <  WT1_3,  WT1_4,  WT1_5,  WT1_9,  WT1_10,
WT1_11 >;

T2 : NEW < T2_7, T2_13 >;

WT2 : NEW < WT2_7, WT2_13 >;

MAIN : SUBR;

  PICK_AND_PLACE : SUBR(DE,PARA,WS,WT);
    PMOVE(WS);
    ZONE(PRECISAO);
    PMOVE(DE);
    DOWN;
    DELAY(ESPERA);
    GRASP;
    DELAY(ESPERA);
    UP;
    ZONE(0);
    PMOVE(WT);
    ZONE(PRECISAO);
    PMOVE(PARA);
    DOWN;
    DELAY(ESPERA);
    RELEASE;
    DELAY(ESPERA);
    UP;
    ZONE(0);
  END;

-- COMANDOS INICIAIS DO ARQUIVO "INITIAL.AML"

UP;
RELEASE;
PMOVE(HOME);
  ITERATE('PICK_AND_PLACE', S1, T1, WS1, WT1);
  ITERATE('PICK_AND_PLACE', S2, T2, WS2, WT2);
END;
```

Figura 5

```
- DEFINICOES DO ARQUIVO DEFAULT.AML :

PRECISAO : NEW 1;
VELOCIDADE : NEW 10;
ESPERA : NEW 1.5;
HOME : NEW PT( 650, 0, 0 );

-- COORDENADAS DOS ALIMENTADORES

S1  : NEW PT(  -93.99, 358.23,-135.22); -- componente1
S2  : NEW PT( -125.31, 410.39,-135.22); -- componente2

-- JA EXISTE UM ALIMENENTADOR PARA O TIPO ABAIXO
-- S2  : NEW PT( -163.64, 448.43,-135.22); -- componente2

-- COORDENADAS DOS COMPONENTES NA PLACA
T1_3  : NEW PT(  -38.00, 575.76, -90.00); -- componente1
T1_4  : NEW PT(  101.87, 573.11,   0.00); -- componente1
T1_5  : NEW PT(  145.30, 548.99,  45.00); -- componente1
T1_9  : NEW PT(  132.57, 293.94,-180.00); -- componente1
T1_10 : NEW PT(   -4.51, 296.59,-180.00); -- componente1
T1_11 : NEW PT(  -47.94, 320.71,-135.00); -- componente1
T2_6  : NEW PT(  -12.31, 577.26,   0.00); -- componente2
T2_13 : NEW PT(   93.27, 292.44,-180.00); -- componente2

-- PONTOS INTERMEDIARIOS PARA ALIMENTADOR E
PLACAS

WT1_3  : NEW PT(  -30.64, 582.53, -90.00); -- componente1
WT1_4  : NEW PT(   97.31, 564.21,   0.00); -- componente1
WT1_5  : NEW PT(  138.31, 541.84,  45.00); -- componente1
WT1_9  : NEW PT(  122.97, 296.74,-180.00); -- componente1
WT1_10 : NEW PT(  -12.23, 302.95,-180.00); -- componente1
WT1_11 : NEW PT(  -54.80, 327.99,-135.00); -- componente1
WS1  : NEW PT(  -87.13, 350.96,-135.22); -- componente1
WT2_6  : NEW PT(  -13.25, 567.30,   0.00); -- componente2
WT2_13 : NEW PT(   84.69, 297.58,-180.00); -- componente2
WS2  : NEW PT( -116.73, 405.26,-135.22); -- componente2

- JA EXISTE UM ALIMENENTADOR PARA O TIPO ABAIXO
-- WS2  : NEW PT( -116.73, 405.26,-135.22); -- componente2
```

```
-MAIN : SUBR;

  PICK_AND_PLACE : SUBR(DE,PARA,WS,WT);
    PMOVE(WS);
    ZONE(PRECISAO);
    PMOVE(DE);
    DOWN;
    DELAY(ESPERA);
    GRASP;
    DELAY(ESPERA);
    UP;
    ZONE(0);
    PMOVE(WT);
    ZONE(PRECISAO);
    PMOVE(PARA);
    DOWN;
    DELAY(ESPERA);
    RELEASE;
    DELAY(ESPERA);
    UP;
    ZONE(0);
  END;

-- COMANDOS INICIAIS DO ARQUIVO "INITIAL.AML"

UP;
RELEASE;
PMOVE(HOME);
  PICK_AND_PLACE( S2 , T2_6, WS2, WT2_6 );
  PICK_AND_PLACE( S1 , T1_3, WS1, WT1_3 );
  PICK_AND_PLACE( S1 , T1_4, WS1, WT1_4 );
  PICK_AND_PLACE( S1 , T1_5, WS1, WT1_5 );
  PICK_AND_PLACE( S2 , T2_13, WS2, WT2_13 );
  PICK_AND_PLACE( S1 , T1_9, WS1, WT1_9 );
  PICK_AND_PLACE( S1 , T1_10, WS1, WT1_10 );
  PICK_AND_PLACE( S1 , T1_11, WS1, WT1_11 );
END;
```

Figura 6

COMENTÁRIOS

O modelamento geométrico é limitado a 2D porque o robô não é servo-assistido no eixo Z, sendo que este eixo é programado apenas para as posições alta ("UP") e baixa ("DOWN").

Uma questão importante é a precisão envolvida (Radhakrishnan 1992), desde que os desenhos CAD são uma representação matemática idealizada da realidade, e temos dificuldade em fazer com que a montagem real coincida com a montagem do lay-out descrito no arquivo CAD.

Além disso temos que levar em conta as caraterísticas de precisão do robô (acurácia e repetibilidade), e compará-las com a exigência de precisão da montagem em questão, e determinarmos se a montagem automática é viável. As caraterísticas do robô podem ser determinadas com testes como os propostos por Edkins e Smith (Edkins M. and Smith C. R. T. 1985), e usar métodos de comparação (Vetorazzi Jr., C. N., 1994) para determinarmos a viabilidade.

No mínimo temos uma boa primeira aproximação tanto do programa quanto das coordenadas, o que pode economizar bastante tempo de programação.

REFERÊNCIAS

(Aut89) "AutoCAD Reference Manual", Autodesk Inc., 1989.

(duF86) du Feu, P.H., "FMS for PCB Assembly", Stanford FMS Proceedings, pp 657-666,Stanford , U.K., 1986.

(Edk85) Edkins, M. and Smith, C. R. T., "The Practical Problems Involved in off-line Programming a Robot from a CAD System", in Robots and Automated

Manufacture, edited by J. Billingsley, P. Peregrinus Ltd., pp 29-39, London, U. K., 1985.

(Jac91) Jacobs, F. et al, "A Rule Based System to Generate NC Programs from CAD Exchange Files", Comp. Ind. Eng., V 20, n. 2, pp 214-3224, 1991.

(Sup91) Supinski, M.R. et al, "Automatic Plan and Robot Code Generation for PCB Assembly", Manufacturing Review V. 4, n. 3, pp 214-224,1991.

(Rad92) Radhakrishnan,T.,"Combined Efects of Linear and Angular Errors in PC Board Assembly", Int. Journal of of Production Research, V. 30, n. 5, pp 1037-1044, 1992.

(Ve94a) Vetorazzi Jr, C. N.,Telles,G. N.,"Robot Program Generation for PCB (Printed Circuit Board) Component Insertion Via CAD System",IEEE ISIE 94 Proceedings,Santiago, Chile, 1994.

(Ve94b) Vetorazzi Jr.,C.N., "Geração Automática de Programas para um Robô Industrial", Dissertação de Mestrado, Faculdade de Engenharia Mecânica - UNICAMP , 1994.