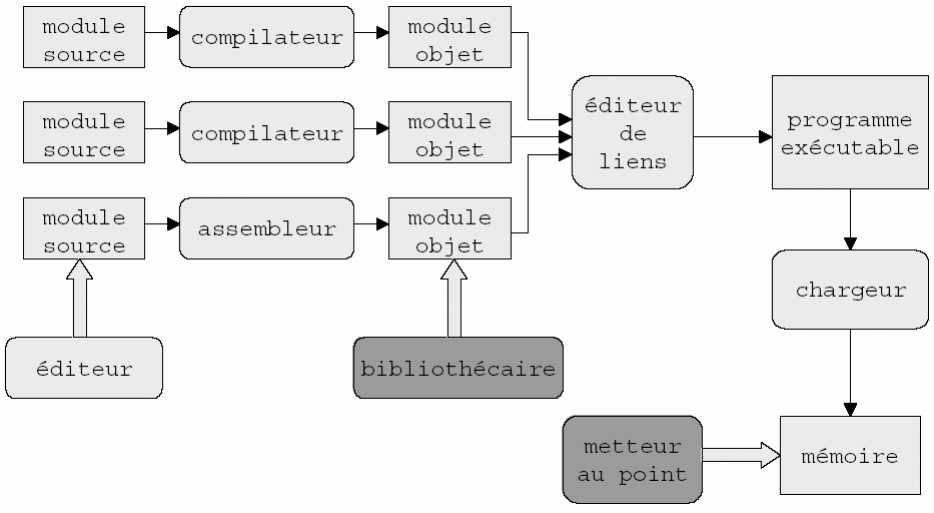
Cours IFT1214 - Automne 2004
Introduction aux systèmes informatiques

Outils de la chaîne de production

Professeur:
Victor Ostromoukhov

Outils de la chaîne de production de programme

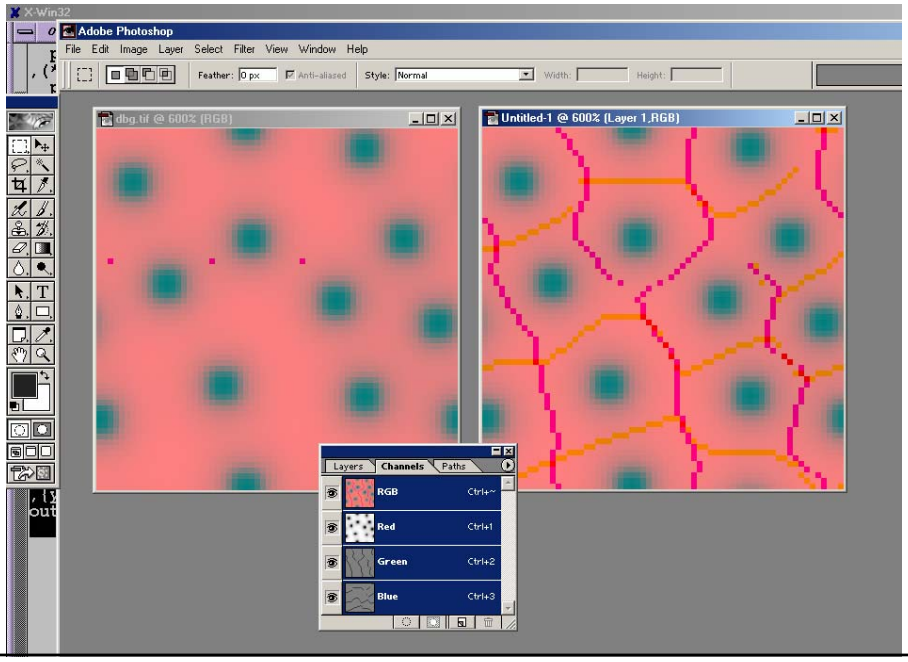


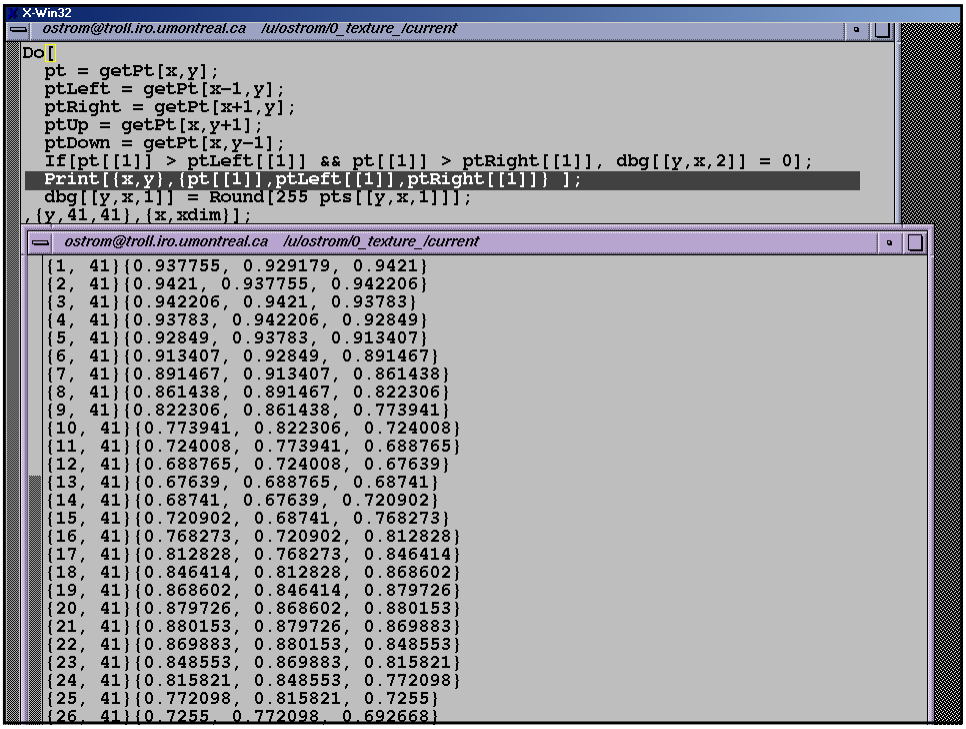
Notion de trace

- Suivre l'évolution de l'état du programme
  - appels de sous programmes, paramètres, retour
  - déroulement des instructions, ruptures de séquences
  - valeurs successives des variables
  - état des données en certains points précis
- Compilation
  - instructions générées par le traducteur
  - difficulté de déterminer les bonnes options (ni trop, ni trop peu)
  - recompilation pour étendre ou supprimer la trace
- Édition de liens ou chargement => pas commode

  => *nécessité d'un outil interactif*

Notion de trace: exemple





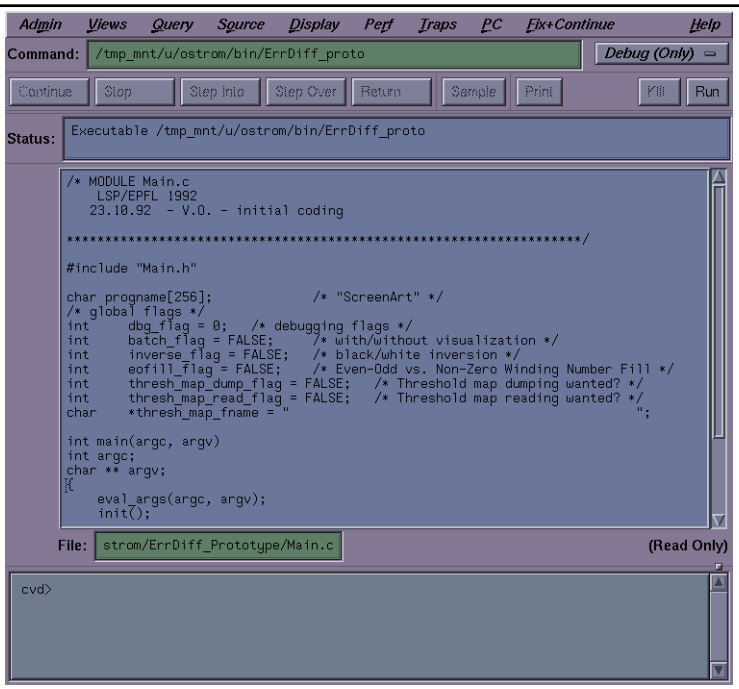
Point d'arrêt, reprise

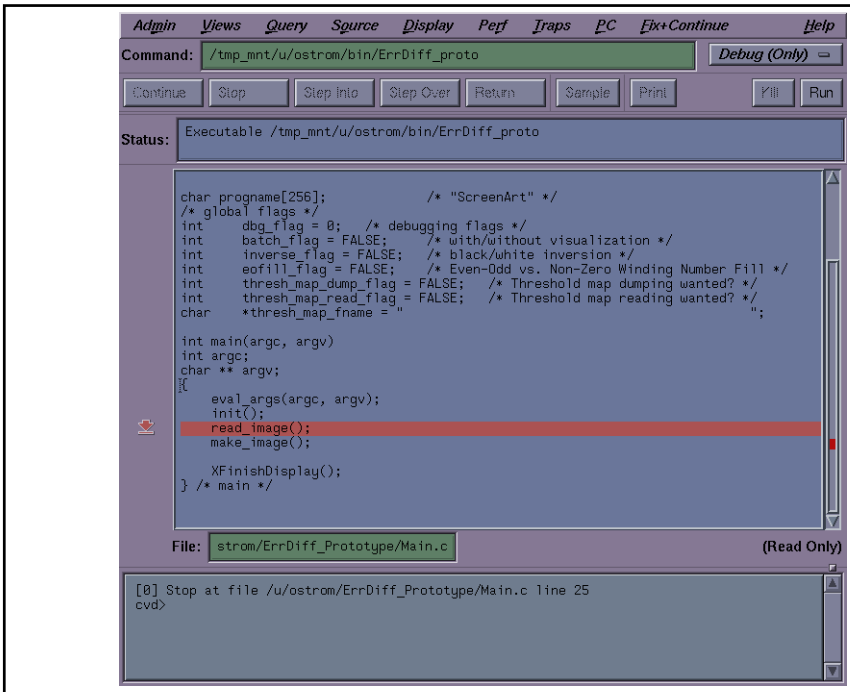
- Notion de point d'arrêt
  - adresse d'emplacement contenant une instruction du programme où on désire que son exécution soit interrompue
  - but: permettre la consultation de l'état du programme à cet endroit
- Notion de reprise
  - permettre de poursuivre l'exécution d'un programme interrompu sur un point d'arrêt
- Notion de pas à pas
  - exécution du programme une instruction à la fois
    - niveau machine => instruction machine et non langage évolué
    - niveau langage évolué => où commence chaque instruction du langage?

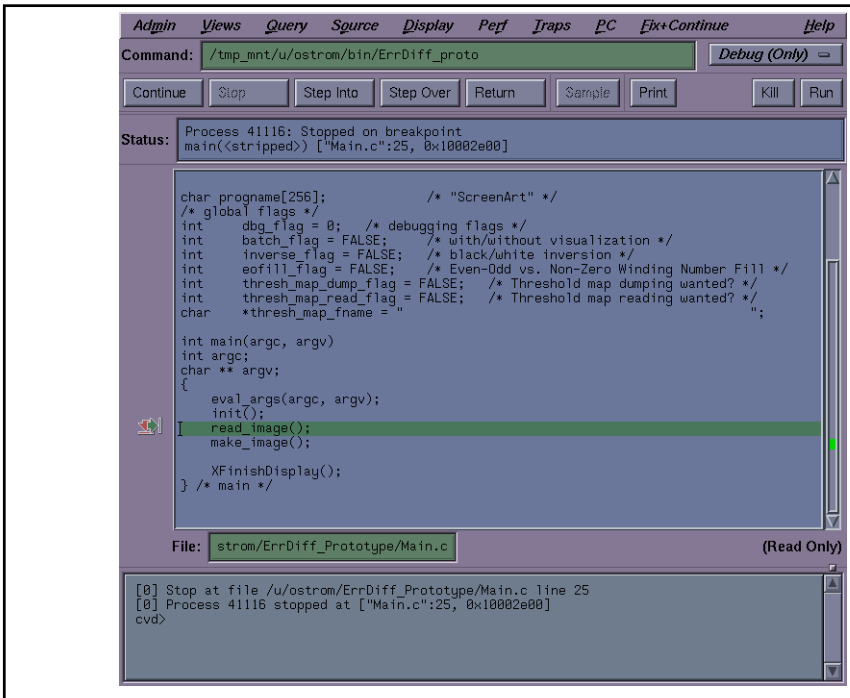
Notion de metteur au point: exemple

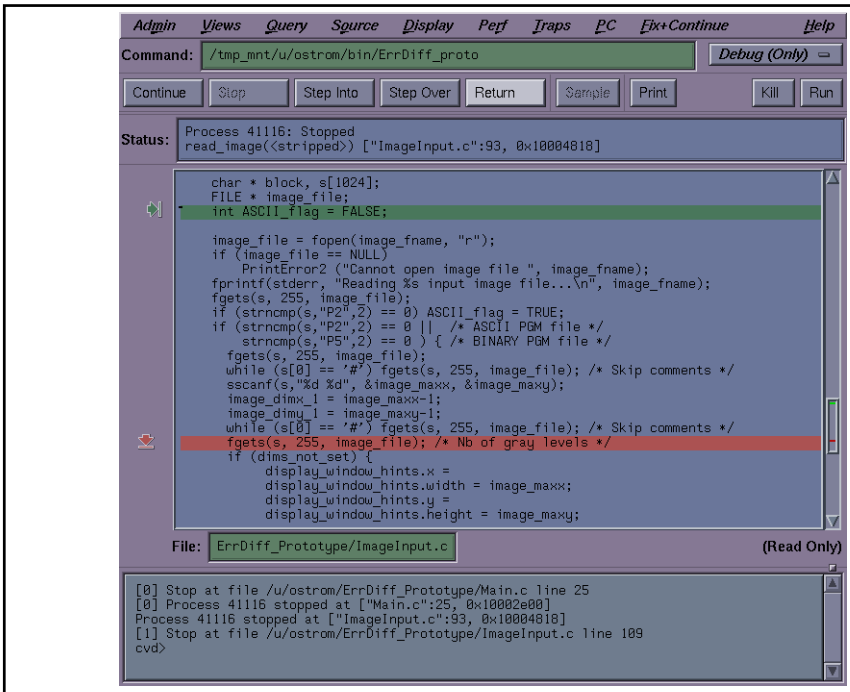
```
> ls
Gamma.c  ImageInput.c  Main.h  XInterface.c  mksamples
Image.c  Main.c        Misc.c  makefile
>
> make
cc -o /u/ostrom/bin/ErrDiff_proto Main.o Image.o ImageInput.o
Gamma.o
XInterface.o Misc.o -lX11 -lm
>
>
> cvd /u/ostrom/bin/ErrDiff_proto
```

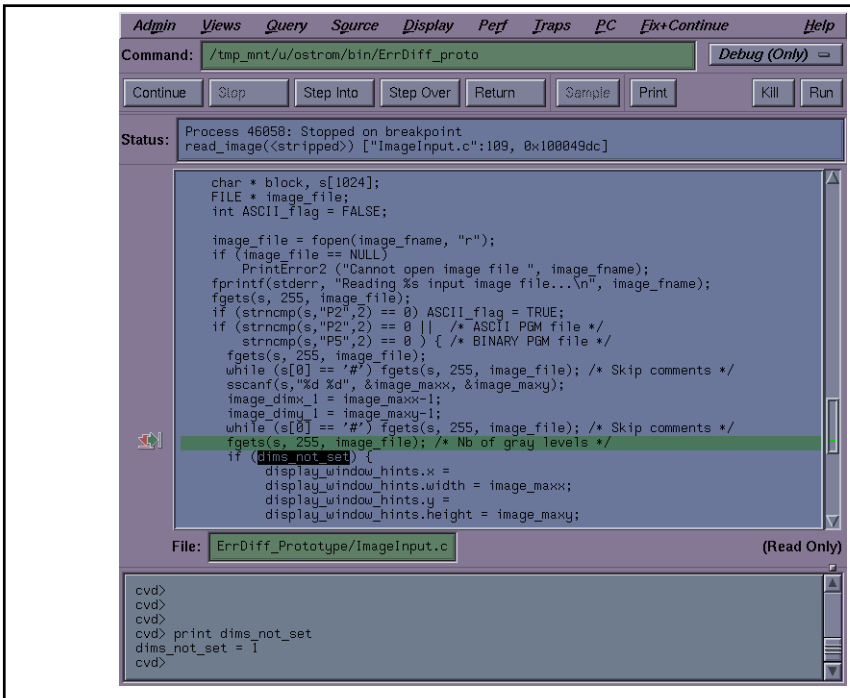
```
> more makefile
# Makefile for ErrDiff_proto generation
#

HOMEDIR = ${HOME}

CFLAGS = -g
COPTFLAGS = -O
COPTFLAGS = -g

COMPILE = cc -c

SOURCES = Main.c\
        Image.c \
        Image.c \
        ImageInput.c \
        Gamma.c \
        XInterface.c \
        Misc.c

OBJECTS = Main.o\
        Image.o \
        ImageInput.o \
        Gamma.o \
        XInterface.o \
        Misc.o

all: ErrDiff_proto

ErrDiff_proto:  ${OBJECTS}
        cc -o ${HOME}/bin/ErrDiff_proto ${OBJECTS} -lX11 -lm

ErrDiff_proto.o:  ErrDiff_proto.c
        ${COMPILE} ${CFLAGS} ErrDiff_proto.c

Main.o: Main.c Main.h
        ${COMPILE} ${OUTPUT_OPTION} $<

ImageInput.o: ImageInput.c Main.h
        ${COMPILE} ${OUTPUT_OPTION} $<

Gamma.o: Gamma.c Main.h
        ${COMPILE} ${OUTPUT_OPTION} $<

Screens.o: Screens.c Main.h
        ${COMPILE} ${OUTPUT_OPTION} $<
```

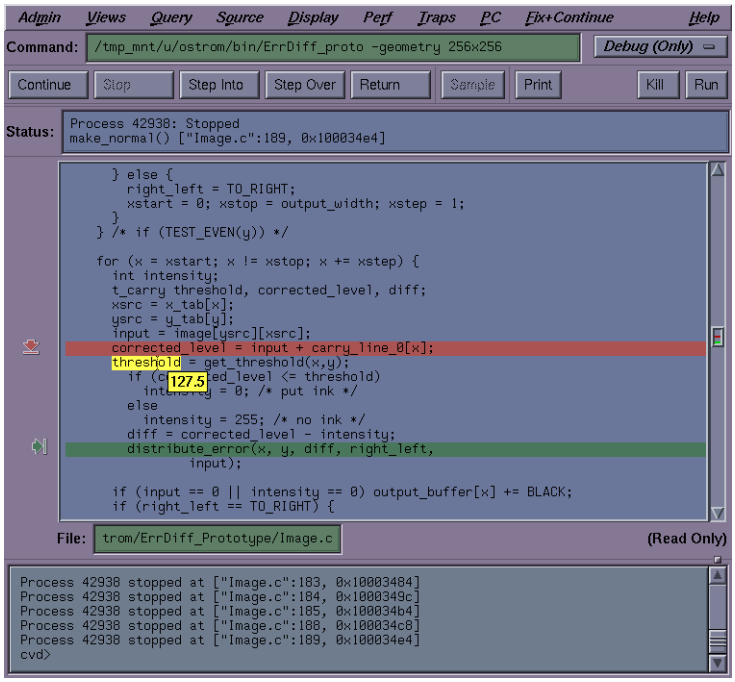


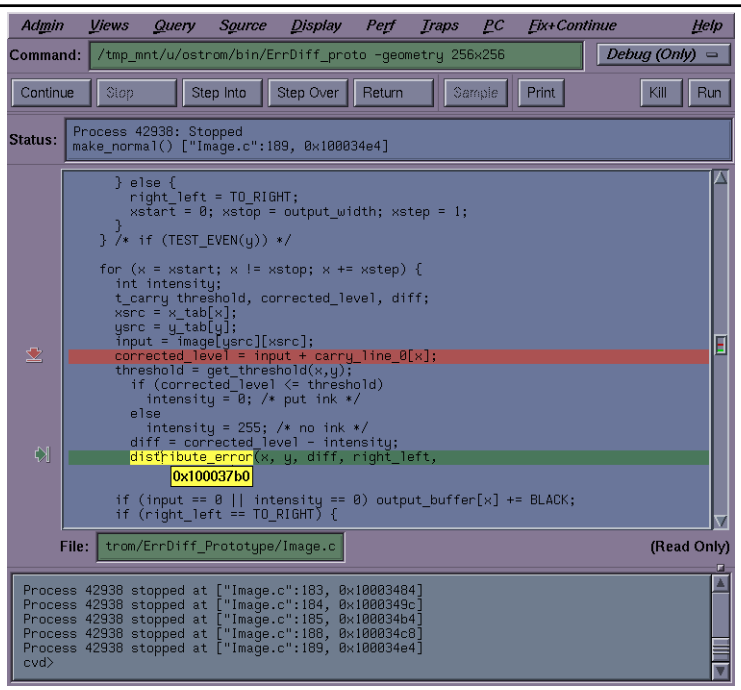

Admin Views Query Source Display Perf Traps PC Fix+Continue Help
Command: /tmp_mnt/u/ostrom/bin/ErrDiff_proto
Debug (Only)
Continue Stop Step Into Step Over Return Sample Print Kill Run
Status: Executable /tmp_mnt/u/ostrom/bin/ErrDiff_proto
char progname[256]; /* "ScreenArt" */
/* global flags */
int dbg_flag = 0; /* debugging flags */
int batch_flag = FALSE; /* with/without visualization */
int inverse_flag = FALSE; /* black/white inversion */
int eofill_flag = FALSE; /* Even-Odd vs. Non-Zero Winding Number Fill */
int thresh_map_dump_flag = FALSE; /* Threshold map dumping wanted? */
int thresh_map_read_flag = FALSE; /* Threshold map reading wanted? */
char *thresh_map_fname = " ";
int main(argc, argv)
int argc;
char ** argv;
{
eval_args(argc, argv);
init();
read_image();
make_image();
XFinishDisplay();
} /* main */
File: strom/ErrDiff_Prototype/Main.c
(Read Only)
[0] Stop at file /u/ostrom/ErrDiff_Prototype/Main.c line 25
cvd>

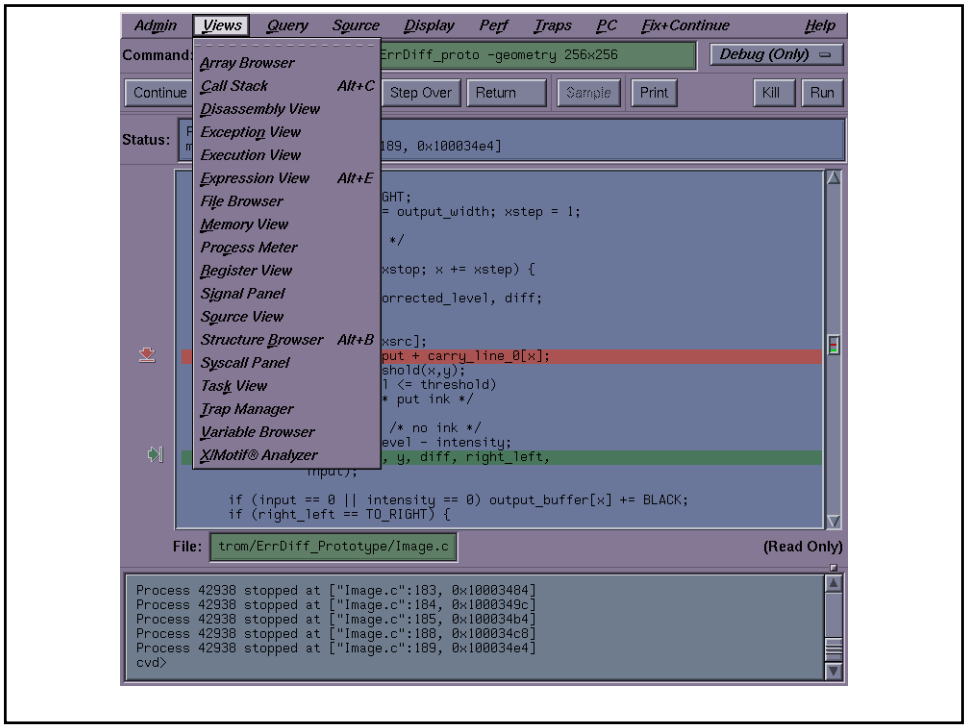

Admin Views Query Source Display Perf Traps PC Fix+Continue Help
Command: /tmp_mnt/u/ostrom/bin/ErrDiff_proto
Debug (Only)
Continue Stop Step Into Step Over Return Sample Print Kill Run
Status: Process 41116: Stopped on breakpoint
main(<stripped>) ["Main.c":25, 0x10002e00]
char progname[256]; /* "ScreenArt" */
/* global flags */
int dbg_flag = 0; /* debugging flags */
int batch_flag = FALSE; /* with/without visualization */
int inverse_flag = FALSE; /* black/white inversion */
int eofill_flag = FALSE; /* Even-Odd vs. Non-Zero Winding Number Fill */
int thresh_map_dump_flag = FALSE; /* Threshold map dumping wanted? */
int thresh_map_read_flag = FALSE; /* Threshold map reading wanted? */
char *thresh_map_fname = " ";
int main(argc, argv)
int argc;
char ** argv;
{
eval_args(argc, argv);
init();
read_image();
make_image();
XFinishDisplay();
} /* main */
File: strom/ErrDiff_Prototype/Main.c
(Read Only)
[0] Stop at file /u/ostrom/ErrDiff_Prototype/Main.c line 25
[0] Process 41116 stopped at ["Main.c":25, 0x10002e00]
cvd>

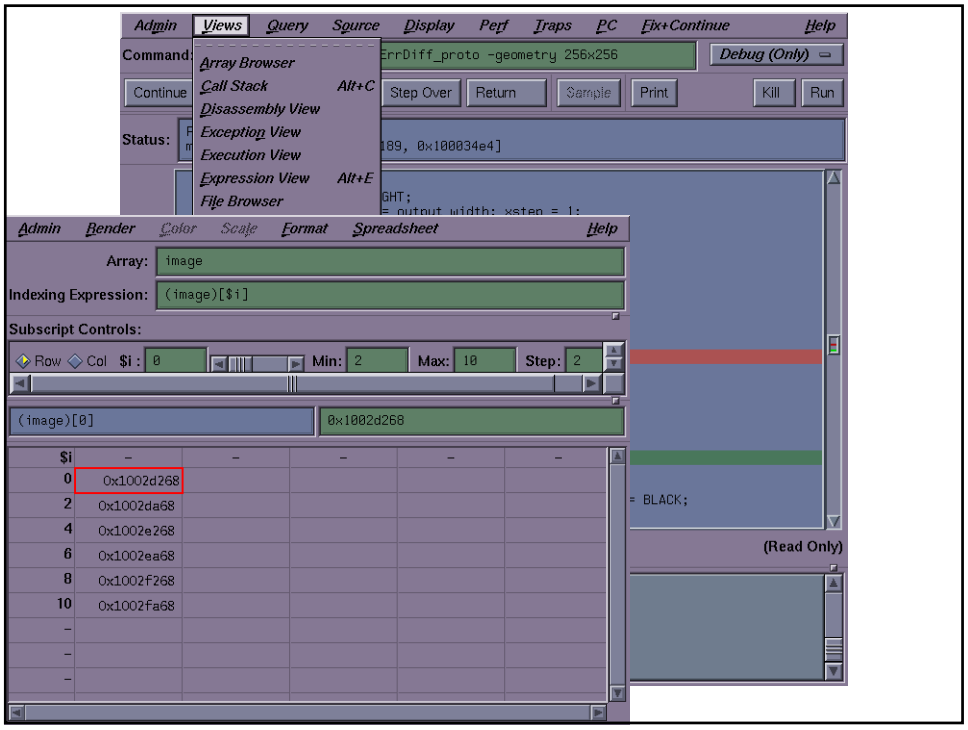

Admin Views Query Source Display Perf Traps PC Fix+Continue Help
Command: /tmp_mnt/u/ostrom/bin/ErrDiff_proto
Debug (Only)
Continue Stop Step Into Step Over Return Sample Print Kill Run
Status: Process 41116: Stopped
read_image(<stripped>) ["ImageInput.c":93, 0x10004818]
char * block, s[1024];
FILE * image_file;
int ASCII_flag = FALSE;
image_file = fopen(image_fname, "r");
if (image_file == NULL)
PrintError2 ("Cannot open image file ", image_fname);
fprintf(stderr, "Reading %s input image file...\n", image_fname);
fgets(s, 255, image_file);
if (strncmp(s,"P2",2) == 0) ASCII_flag = TRUE;
if (strncmp(s,"P2",2) == 0 || /* ASCII PGM file */
strncmp(s,"P5",2) == 0) { /* BINARY PGM file */
fgets(s, 255, image_file);
while (s[0] == '#') fgets(s, 255, image_file); /* Skip comments */
sscanf(s,"%d %d", &image_maxx, &image_maxy);
image_dimx_1 = image_maxx-1;
image_dimy_1 = image_maxy-1;
while (s[0] == '#') fgets(s, 255, image_file); /* Skip comments */
fgets(s, 255, image_file); /* Nb of gray levels */
if (dims_not_set) {
display_window_hints.x =
display_window_hints.width = image_maxx;
display_window_hints.y =
display_window_hints.height = image_maxy;
File: ErrDiff_Prototype/ImageInput.c
(Read Only)
[0] Stop at file /u/ostrom/ErrDiff_Prototype/Main.c line 25
[0] Process 41116 stopped at ["Main.c":25, 0x10002e00]
Process 41116 stopped at ["ImageInput.c":93, 0x10004818]
[1] Stop at file /u/ostrom/ErrDiff_Prototype/ImageInput.c line 109
cvd>

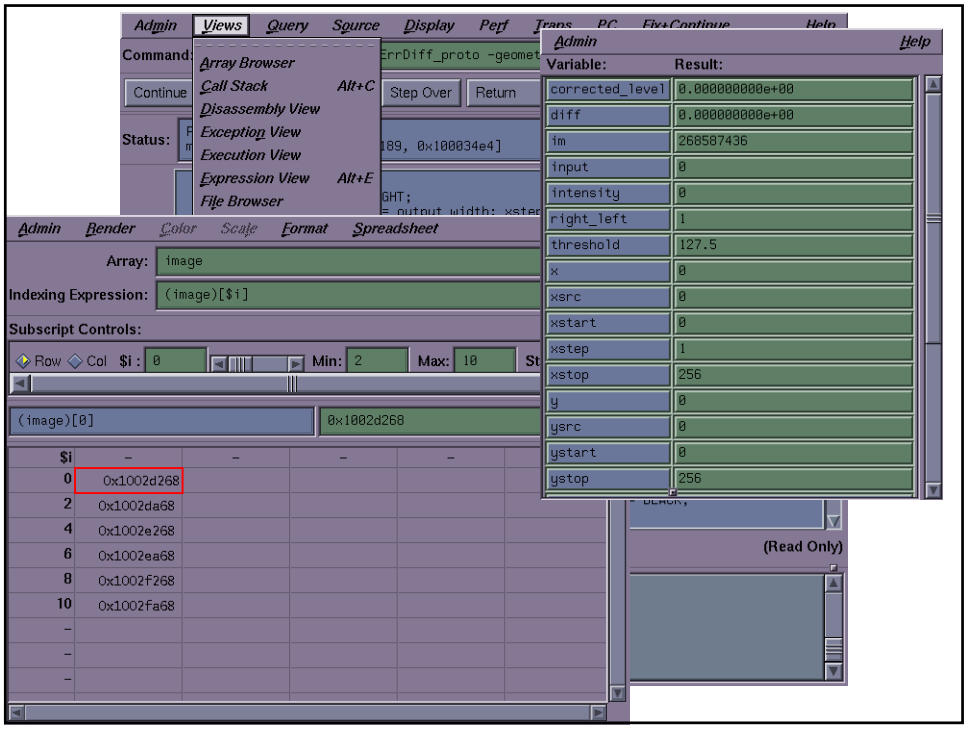

Admin Views Query Source Display Perf Traps PC Fix+Continue Help
Command: /tmp_mnt/u/ostrom/bin/ErrDiff_proto
Debug (Only)
Continue Stop Step Into Step Over Return Sample Print Kill Run
Status: Process 46058: Stopped on breakpoint
read_image(<stripped>) ["ImageInput.c":109, 0x100049dc]
char * block, s[1024];
FILE * image_file;
int ASCII_flag = FALSE;
image_file = fopen(image_fname, "r");
if (image_file == NULL)
PrintError2 ("Cannot open image file ", image_fname);
fprintf(stderr, "Reading %s input image file...\n", image_fname);
fgets(s, 255, image_file);
if (strncmp(s,"P2",2) == 0) ASCII_flag = TRUE;
if (strncmp(s,"P2",2) == 0 || /* ASCII PGM file */
strncmp(s,"P5",2) == 0) { /* BINARY PGM file */
fgets(s, 255, image_file);
while (s[0] == '#') fgets(s, 255, image_file); /* Skip comments */
sscanf(s,"%d %d", &image_maxx, &image_maxy);
image_dimx_1 = image_maxx-1;
image_dimy_1 = image_maxy-1;
while (s[0] == '#') fgets(s, 255, image_file); /* Skip comments */
fgets(s, 255, image_file); /* Nb of gray levels */
if (dims_not_set) {
display_window_hints.x =
display_window_hints.width = image_maxx;
display_window_hints.y =
display_window_hints.height = image_maxy;
File: ErrDiff_Prototype/ImageInput.c
(Read Only)
cvd>
cvd>
cvd>
cvd> print dims_not_set
dims_not_set = 1
cvd>

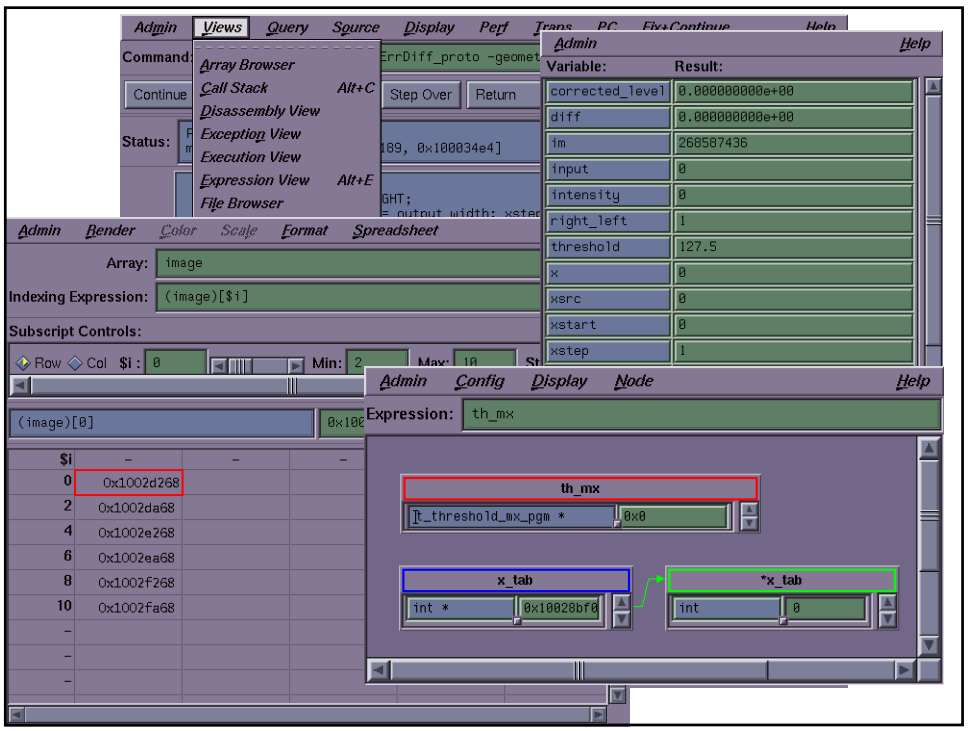

Admin Views Query Source Display Perf Traps PC Fix+Continue Help
Command: /tmp_mnt/u/ostrom/bin/ErrDiff_proto -geometry 256x256
Debug (Only)
Continue Stop Step Into Step Over Return Sample Print Kill Run
Status: Process 42938: Stopped
make_normal() ["Image.c":189, 0x100034e4]
} else {
right_left = TO_RIGHT;
xstart = 0; xstop = output_width; xstep = 1;
}
} /* if (TEST_EVEN(y)) */
for (x = xstart; x != xstop; x += xstep) {
int intensity;
t_carry threshold, corrected_level, diff;
xsrc = x_tab[x];
ysrc = y_tab[y];
input = image[ysrc][xsrc];
corrected_level = input + carry_line_0[x];
threshold = get_threshold(x,y);
if (corrected_level <= threshold)
127.5
intensity = 0; /* put ink */
else
intensity = 255; /* no ink */
diff = corrected_level - intensity;
distribute_error(x, y, diff, right_left,
input);
if (input == 0 || intensity == 0) output_buffer[x] += BLACK;
if (right_left == TO_RIGHT) {
File: trom/ErrDiff_Prototype/Image.c
(Read Only)
Process 42938 stopped at ["Image.c":183, 0x10003484]
Process 42938 stopped at ["Image.c":184, 0x1000349c]
Process 42938 stopped at ["Image.c":185, 0x100034b4]
Process 42938 stopped at ["Image.c":188, 0x100034c8]
Process 42938 stopped at ["Image.c":189, 0x100034e4]
cvd>

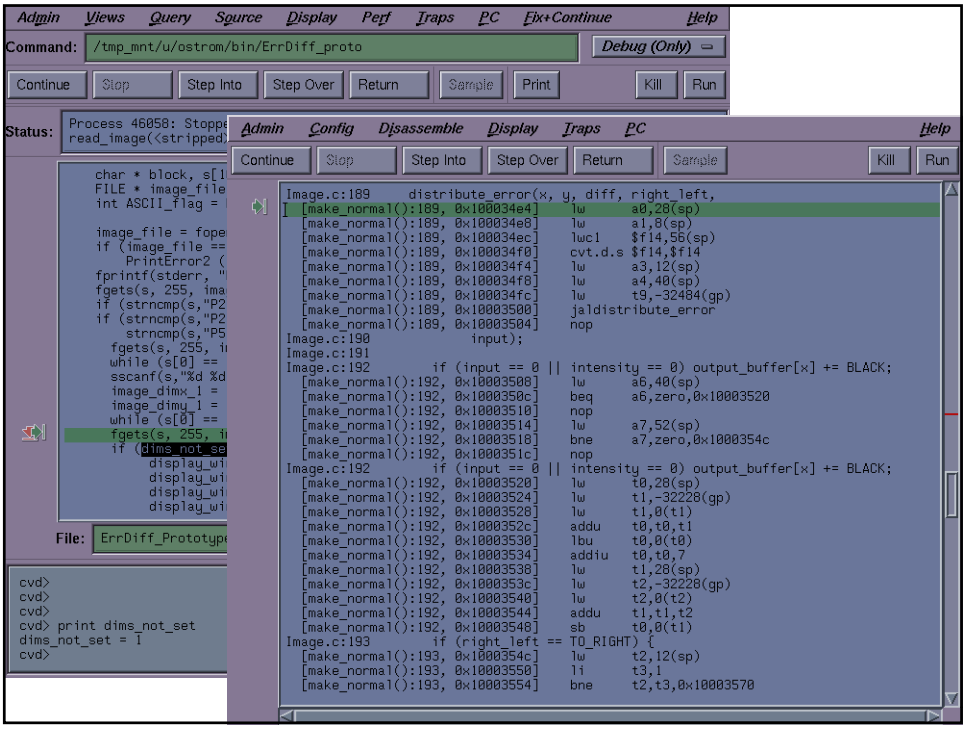

Admin Views Query Source Display Perf Traps PC Fix+Continue Help
Command: /tmp_mnt/u/ostrom/bin/ErrDiff_proto -geometry 256x256
Debug (Only)
Continue Stop Step Into Step Over Return Sample Print Kill Run
Status: Process 42938: Stopped
make_normal() ["Image.c":189, 0x100034e4]
} else {
right_left = TO_RIGHT;
xstart = 0; xstop = output_width; xstep = 1;
}
} /* if (TEST_EVEN(y)) */
for (x = xstart; x != xstop; x += xstep) {
int intensity;
t_carry threshold, corrected_level, diff;
xsrc = x_tab[x];
ysrc = y_tab[y];
input = image[ysrc][xsrc];
corrected_level = input + carry_line_0[x];
threshold = get_threshold(x,y);
if (corrected_level <= threshold)
intensity = 0; /* put ink */
else
intensity = 255; /* no ink */
diff = corrected_level - intensity;
distribute_error(x, y, diff, right_left,
0x100037b0
if (input == 0 || intensity == 0) output_buffer[x] += BLACK;
if (right_left == TO_RIGHT) {
File: trom/ErrDiff_Prototype/Image.c
(Read Only)
Process 42938 stopped at ["Image.c":183, 0x10003484]
Process 42938 stopped at ["Image.c":184, 0x1000349c]
Process 42938 stopped at ["Image.c":185, 0x100034b4]
Process 42938 stopped at ["Image.c":188, 0x100034c8]
Process 42938 stopped at ["Image.c":189, 0x100034e4]
cvd>


Admin
Views
Query
Source
Display
Perf
Traps
PC
Fix+Continue
Help
Command:
ErrDiff_proto -geometry 256x256
Debug (Only)
Continue
Step Over
Return
Sample
Print
Kill
Run
Status:
Array Browser
Call Stack Alt+C
Disassembly View
Exception View
Execution View
Expression View Alt+E
File Browser
Memory View
Process Meter
Register View
Signal Panel
Source View
Structure Browser Alt+B
Syscall Panel
Task View
Trap Manager
Variable Browser
X/Motif® Analyzer
File:
trom/ErrDiff_Prototype/Image.c
(Read Only)
Process 42938 stopped at ["Image.c":183, 0x10003484]
Process 42938 stopped at ["Image.c":184, 0x1000349c]
Process 42938 stopped at ["Image.c":185, 0x100034b4]
Process 42938 stopped at ["Image.c":188, 0x100034c8]
Process 42938 stopped at ["Image.c":189, 0x100034e4]
cvd>


Admin
Render
Color
Scale
Format
Spreadsheet
Help
Array:
image
Indexing Expression:
(image)[$i]
Subscript Controls:
Row
Col
$i :
0
Min:
2
Max:
10
Step:
2
(image)[0]
0x1002d268
$i
0
2
4
6
8
10
0x1002d268
0x1002da68
0x1002e268
0x1002ea68
0x1002f268
0x1002fa68
(Read Only)


Admin
Views
Query
Source
Display
Perf
Traps
PC
Fix+Continue
Help
Command:
Array Browser
Call Stack Alt+C
Disassembly View
Exception View
Execution View
Expression View Alt+E
File Browser
Continue
Step Over
Return
Status:
Admin
Bender
Color
Scale
Format
Spreadsheet
Array: image
Indexing Expression: (image)[$i]
Subscript Controls:
Row
Col
$i :
Min:
Max:
(image)[0]
0x1002d268
$i
0 0x1002d268
2 0x1002da68
4 0x1002e268
6 0x1002ea68
8 0x1002f268
10 0x1002fa68
Admin
Help
Variable:
Result:
corrected_level 0.000000000e+00
diff 0.000000000e+00
im 268587436
input 0
intensity 0
right_left 1
threshold 127.5
x 0
xsrc 0
xstart 0
xstep 1
xstop 256
y 0
ysrc 0
ystart 0
ystop 256
(Read Only)


Admin
Views
Query
Source
Display
Perf
Traps
PC
Fix+Continue
Help
Command:
Array Browser
Call Stack Alt+C
Disassembly View
Exception View
Execution View
Expression View Alt+E
File Browser
Continue
Step Over
Return
Status:
Admin
Bender
Color
Scale
Format
Spreadsheet
Array: image
Indexing Expression: (image)[$i]
Subscript Controls:
Row
Col
$i :
Min:
(image)[0]
$i
0 0x1002d268
2 0x1002da68
4 0x1002e268
6 0x1002ea68
8 0x1002f268
10 0x1002fa68
Admin
Help
Variable:
Result:
corrected_level 0.000000000e+00
diff 0.000000000e+00
im 268587436
input 0
intensity 0
right_left 1
threshold 127.5
x 0
xsrc 0
xstart 0
xstep 1
Admin
Config
Display
Node
Help
Expression: th_mx
th_mx
_t_threshold_mx_pgm *
0x0
x_tab
int *
0x10028bf0
*x_tab
int
0



Notion de metteur au point

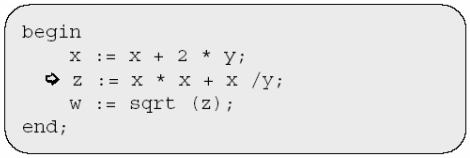
- outil interactif qui permet:
  - mettre en route ou arrêter des options de trace
  - consulter et modifier des variables du programme
  - créer et supprimer des points d'arrêt
  - exécuter le programme en pas à pas
- metteur au point binaire
  - fournit ces fonctionnalités au niveau langage machine
    - variables => adresses binaires des emplacements
      format d'édition par décision de l'utilisateur
    - instructions => celles de la machine binaire
      avec éventuellement désassembleur (mnémonique)
  - nécessite une bonne connaissance de l'architecture de la machine

Metteur au point symbolique

- Fonctionnalités habituelles
  - mise en route et arrêt des traces, consultation et modification des variables création et suppression des points d'arrêts, exécution en pas à pas
- En plus:
  - consultation des appels de procédures en cours
  - consultation du fichier source
  - édition du fichier source (aide mémoire)
- Surtout: connaissance du langage source
  - identificateurs du texte source et adresses associées
  - déclarations et donc types des objets => format et notation pointée
  - correspondance instruction source -> emplacements mémoire
- Par le biais de tables
  traducteurs -> éditeur de liens -> metteur au point

Metteur au point multi-fenêtre

fenêtre de source:
point d'arrêt actuel



fenêtre des variables:

boutons de commande:

fenêtre de commandes:

Mesures de comportement dynamique

- Rôle:
  - compter le nombre de fois où une suite d'instructions est exécutée
  - mesurer le temps passé à l'exécution dans une suite d'instructions
- Moyen:
  - option du traducteur (insertion d'instructions machines)
  - incrémentation d'un compteur avant l'exécution
  - appel au système avant et après l'exécution
- Résultats:
  - mémorisés dans un fichier et rattachés au source par un outil spécifique
- Intérêt:
  - optimiser l'exécution du programme là où on passe beaucoup de temps
- Exemples: en µs => instructions, en ms => sous-programmes

Préprocesseur et macrogénérateur(1)

- Paramétrage de textes sources:
  - texte unique => plusieurs versions de modules objets
- Exemple: préprocesseur C (et Pascal sur Unix, ...)

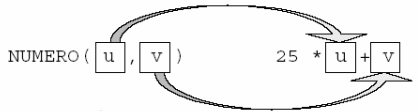
`#include "nom de fichier"`	inclusion du fichier à cet endroit
`#define TOTO 2456`	remplacement de la chaîne `TOTO` par la chaîne `2456`, partout
`#undef TOTO`	arrêt du remplacement de la chaîne `TOTO`
`#ifdef TOTO`	s'il y a un remplacement de `TOTO`, transmettre
`...`	les lignes qui suivent et supprimer celles
`#else`	après `#else`
`...`	sinon, supprimer celles qui suivent et transmettre
`#undef TOTO`	celles après `#else`

• **D'autres exemples:**

```
#define ABS(x) (((x) >= 0) ? (x) : (-(x)))
#define MIN(x,y) ((x) < (y) ? (x) : (y))
#define MAX(x,y) ((x) < (y) ? (y) : (x))
#define ISWAP(x,y) {(x) ^= (y); (y) ^= (x); (x) ^= (y); }
```

Préprocesseur et macrogénérateur(2)

- Exemple: `#define NUMERO(A,B) 25 * A + B`




```
NUMERO(x + 1, y)        25 * x + 1 + y
NUMERO((x + 1), y)      25 * (x + 1) + y
➪ #define NUMERO(A,B) 25 * (A) + (B)
```

- méta-langage: langage de commandes dans un texte source
- macrogénérateur: programme qui interprète ce métalangage pour générer un programme source
- Application:
  - requêtes de bases de données => appels sous programmes

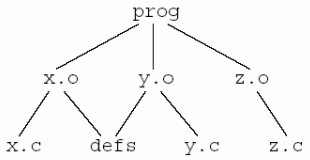
Le Make (1)

- Notion de graphe de dépendance

  On dit qu'un fichier A dépend d'un fichier B si B est utilisé pour construire A
- Si A dépend de B et date(A) < date(B) => A n'est pas à jour
- Exemple:
  - un module objet dépend du module source
  - un programme exécutable dépend des modules objets dont il est l'édition de liens

➪ Graphe de dépendance

`x.c` et `y.c` "incluent" `defs`



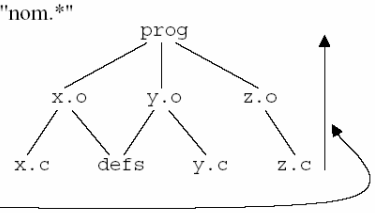
Le Make (2)

- Construction et exploitation du graphe de dépendance
- Définitions explicites dans un fichier paramètre (makefile)

```
prog: x.o y.o z.o
    ld x.o y.o z.o -ls -o prog
x.o y.o: defs
```

- Définitions implicites par suffixes
  - fichier "nom.o", recherche fichiers "nom.*"

```
x.o: x.c
    cc -c x.c
y.o: y.c
    cc -c y.c
z.o: z.c
    cc -c z.c
```



commandes de reconstruction

Le Make (3)

définition de macros

```
OBJECTS = x.o y.o z.o
FILES = x.c y.c z.c defs
LIBES = -ls
P = imprint -Plplaser
```

`make prog`

référence à une macro pour dépendances et commandes

```
prog: $(OBJECTS)
  $(LD) $(OBJECTS) $(LIBES) -o prog
```

paramétrisation éditeur de liens

```
x.o y.o: defs
```

`make print`

`print`: *fichier existant, vide*

```
print: $(FILES)
  $P $?
  touch print
```

`$?`: *les noms de fichiers les plus récents*

modification date de modification

`make arch`

```
arch:
  ar uv /usr/src/prog.a $(FILES)
```

pas de fichier `arch` => exécution

Outils complémentaires

- Archiveur
  - regrouper en un seul fichier plusieurs fichiers quelconques, en les compactant (gain 2 ou plus), avec conservation du découpage
- Différence
  - déterminer la suite de commande éditeur de texte qu'il faut appliquer à un fichier A pour obtenir le fichier B, à partir du contenu de ces fichiers
- Paragrapheur
  - mettre en forme standard un fichier source écrit dans un langage évolué
- Bibliothécaire
  - gérer les modules objets d'une bibliothèque, pour faciliter le travail de l'éditeur de liens

Il est préférable de faire un outil pour une activité automatisable plutôt que de réaliser l'activité à la main, car l'outil sera réutilisable.

Conclusion

- Metteur au point => outil de contrôle d'exécution d'un programme
  - trace: suivi de l'état
  - point d'arrêt: endroit où on force l'arrêt
  - reprise: poursuite d'exécution
  - pas à pas: exécution une instruction à la fois
- Mesures de comportement dynamique: amélioration temps exécution
- Préprocesseur, macrogénérateur : traitement du texte source
- "make": mise à jour automatique des fichiers
  - graphe de dépendance entre des fichiers
  - dates de dernière mise à jour conformes au graphe
  - exécution automatique de commandes si ce n'est pas le cas
- Beaucoup d'outils pour des activités automatisables spécifiques

Résumé

+ La trace d'un programme est une suite d'informations qui permettent de contrôler l'évolution de l'état d'un programme au cours de son exécution. En fait on ne cherche bien souvent à connaître qu'une partie de cette trace.

+ Un point d'arrêt dans un programme est un endroit où l'on désire arrêter l'exécution du programme pour consulter son état interne. La reprise est la possibilité de poursuivre l'exécution du programme après qu'il ait été interrompu. Le pas à pas est la possibilité d'exécuter le programme une instruction à la fois.

+ Un metteur au point est un outil qui permet de contrôler interactivement l'exécution d'un programme. Il est symbolique lorsqu'il permet l'expression des commandes en utilisant les structures du langage évolué dans lequel le programme a été écrit.

+ Les mesures de comportement dynamique d'un programme permettent de savoir où il faut porter son effort pour améliorer son temps d'exécution.

+ Un préprocesseur ou un macrogénérateur est un programme qui permet de faire un traitement simple sur le texte source avant de le donner à un traducteur.

+ Il existe un graphe de dépendance entre l'ensemble des fichiers utilisés dans la chaîne de production de programmes: un fichier A dépend d'un fichier B si B est utilisé pour construire A.

+ Un fichier contenant un programme exécutable est à jour si pour tous les fichiers A et B de son graphe de dépendance, tels que A dépende de B, la date de dernière modification de A est postérieure à celle de B.

+ Le make est un outil qui, à l'aide d'un fichier parmètre, construit le graphe de dépendance d'un programme exécutable, contrôle qu'il est à jour et exécute les commandes minimales pour assurer cette mise à jour, si ce n'est pas le cas.

+ Le make peut être utilisé chaque fois qu'il existe un graphe de dépendance entre des fichiers, pour contrôler que ces fichiers sont à jour, et exécuter les commandes nécessaires à cette mise à jour, si ce n'est pas le cas.

+ Beaucoup d'autres outils existent, car il est souvent préférable de faire un outil pour une activité automatisable, plutôt que de réaliser l'activité à la main.