Recherche reproductible avec Lepton

Li-Thiao-Té Sébastien

LAGA UMR 7539, Université Paris 13

Le logiciel Lepton

- OCaml
- Site web : `http://www.math.univ-paris13.fr/~lithiao/ResearchLepton/Lepton.html`
- avec le programme en libre téléchargement pour Linux 32/64, (Windows)
- manuel + faq + exemples
- 2 conference papers
  - Sébastien Li-Thiao-Té. Literate program execution for reproducible research and executable papers. *Procedia Computer Science*, 9(0) :439 – 448, 2012. ICCS 2012.
  - Sébastien Li-Thiao-Té. Literate program execution for teaching computational science. *Procedia Computer Science*, 9(0) :1723 – 1732, 2012. ICCS 2012.

Fonctionnalités

Lepton est un programme qui traite des blocs de texte

Code chunk 1: ≪lepton≫

```
<<name options>>=
code
@
```

avec quatre opérations de base :

- `-write` permet d'écrire sur le disque
- `-exec interpreter` permet d'exécuter le code
- `-chunk format -output format` contrôle le format de sortie
- `<<chunk_ref>>` permet de réutiliser un autre bloc

Une histoire vécue (en cours)

Code chunk 2: ≪mail≫

```
Date: Thu, 21 Mar 2013 12:37:56 +0000 (GMT)
From: ********
Subject: code
To: Sebastien <lithiao@math.univ-paris13.fr>
voici donc le code demande=20

[...]

Content-Type: text/x-csrc; name="modprogr.c"
```

Une histoire vécue (en cours)

```
/*************************************************************************/
/*        APPLICATION FOR COMPRESSION                                    */
/*************************************************************************/

//main definitions

#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#define max(x,y)        ((x)>(y)?(x):(y))
#define min(x,y)        ((x)<(y)?(x):(y))
#define NR_END 1

/*.....................................................................*/
/*.....................tools....functions..............................*/
/*.....................................................................*/
void nrerror(char error_text[])
{
 /* print an error message*/
fprintf(stderr,"Numerical  run-time error...\n");
fprintf(stderr,"%s\n",error_text);
fprintf(stderr,"...now exiting to system...\n");
exit(1);
}
/*.....................................................................*/
float *vector(long nl, long nh)
/* allocate a float vector with subscript range v[nl..nh] */
{
float *v;

v=(float *)malloc((size_t) ((nh-nl+1+NR_END)*sizeof(float)));
if (!v) nrerror("allocation failure in vector()");
*(v-nl+NR_END)=10.;
return v-nl+NR_END;

}

/*.....................................................................*/

float **matrix(long nrl, long nrh, long ncl, long nch)
/* allocate a float matrix with subscript range m[nrl..nrh][ncl..nch] */
{
long i, nrow=nrh-nrl+1,ncol=nch-ncl+1;
float **m;

/* allocate pointers to rows */
m=(float **) malloc((size_t)((nrow+NR_END)*sizeof(float*)));
if (!m) nrerror("allocation failure 1 in matrix()");
m += NR_END;
m -= nrl;

/* allocate rows and set pointers to them */
m[nrl]=(float *) malloc((size_t)((nrow*ncol+NR_END)*sizeof(float)));
if (!m[nrl]) nrerror("allocation failure 2 in matrix()");
```

modprogr.c

- 5069 lignes de code
- des commentaires mais pas de documentation
- pas de manuel d'installation
- pas de manuel d'utilisation
- 5 méthodes dans le même programme
- paramètres à taper à la main
- redondances
- etc.

Que faire ?

- utiliser le programme
- comprendre le code
- comprendre les méthodes
- adapter le code pour votre propre projet
- garder une trace de votre travail

1e étape : prendre des notes

Fichier Lepton en syntaxe LaTeX avec du code source C.

Code chunk 3: ≪modprogr.nw≫

```
\documentclass[a4paper,10pt]{scrartcl}
\input{lepton.sty}
\usepackage{hyperref}

\begin{document} %
\title{Source code documentation}
\author{Li-Thiao-Te Sebastien}
\maketitle

\section{Compilation}

<<modprogr.c -write>>=
[copier-coller]
@
```

1e étape : prendre des notes

Fichier Lepton en syntaxe LaTeX avec du code source C.

Code chunk 4: «utilisation»

```
lepton modprogr.nw
pdflatex --shell-escape modprogr.tex
```

Résultats :

- le fichier `modprogr.c` est extrait du fichier `modprogr.nw`
- le fichier `modprogr.pdf` contient la documentation

2e étape : compilation

On ajoute les instructions de compilation dans le fichier Lepton.

```
\section{Compilation}

<<modprogr.c -write>>=
[copier-coller]
@
<<shell -exec shell>>=
dpkg -l gcc
gcc -lm modprogr.c -o modprogr.bin
@
```

Code chunk 5: ≪shell (part 2)≫

```
dpkg -l gcc
gcc -lm modprogr.c -o modprogr.bin
```

Interpret with shell

```
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name                                  Version                       Architecture Descripti
+++-=====================================-=============================-============-=========
ii  gcc                                   4:4.7.2-1                     i386         GNU C com
```

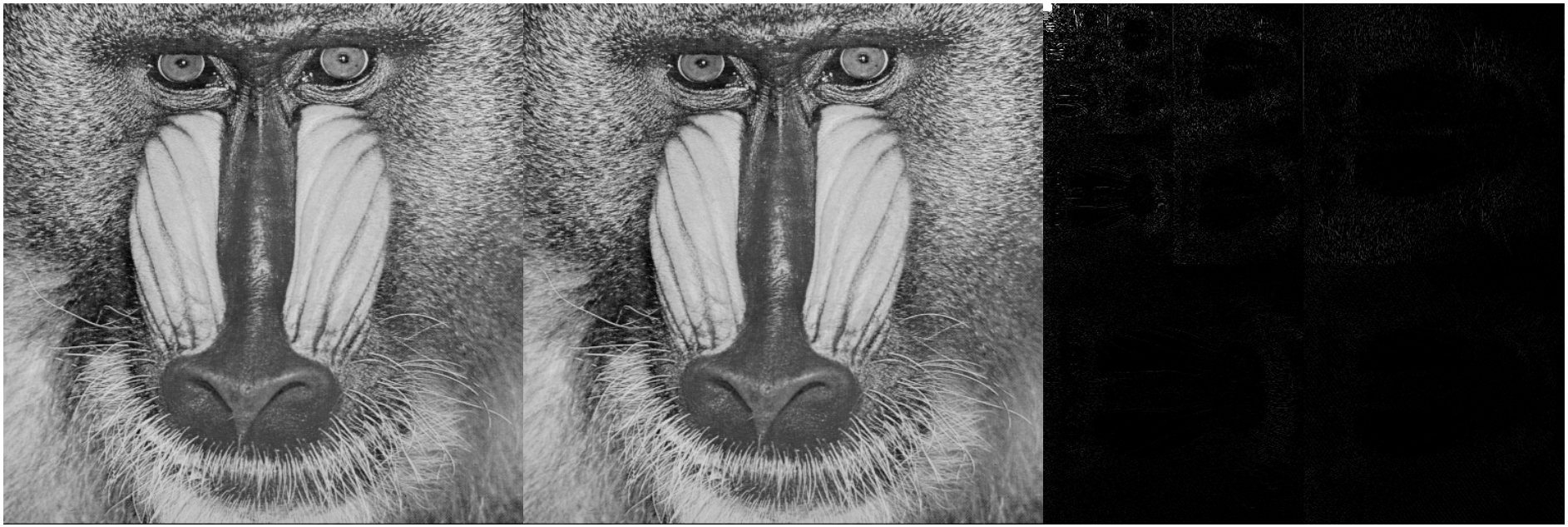
3e étape : utilisation

On teste l'exécutable.

```
            Application for compression/approximation : choose the method

 Linear     Tensor Product Approach  (1)

 ENO        Tensor Product Approach  (2)

 ENO-SR     Tensor Product Approach  (3)

 ENO    NON Tensor Product Approach  (4)

 ENO-SR NON Tensor Product Approach  (5)

1
Original Image ?fig.png
Threshold = 0
Finer level = 8
Lower level = 3
Type of normalization L1 (1), L2 (2) or Linf (3)1
Solution ?sol.png
Decomposition  ?dec.png
lecture ...
fish: Job 1, \./modprogr.bin " terminated by signal SIGSEGV (Address boundary error)
```

C'est le moment d'aller poser une question...

Après discussion



Documentation du code source

Qu'est-ce que la documentation d'un code source ?

- description de l'algorithme
- commentaires dans le code
- structure du code

Pb = respecter la syntaxe du langage de programmation

- ordre de définition des fonctions
- commentaires en texte brut
- ou extraction automatique des commentaires

Description de l'algorithme

```
\section{Compilation}

<<modprogr.c -write>>=
[copier-coller]
@
<<shell -exec shell>>=
dpkg -l gcc
gcc -lm modprogr.c -o modprogr.bin
@

\section{Algorithme}

(A faire!!!)
```

Description de l'algorithme

Faire appel à un programme externe :

- graphe de dépendance
- analyse statique
- etc.

Structure du programme

Code chunk 6: ≪modprogr.c≫

include useful bigfile.c main

```
<<include>>
<<useful>>
<<bigfile.c>>
<<main>>
```

La fonction main demande à l'utilisateur la méthode, puis lance le sous-programme correspondant.

Code chunk 7: ≪main≫

```
main() {
  int ch;
  printf("\n            Application for compression/approximation : choose the method       \n ");
  printf("\n Linear     Tensor Product Approach  (1)\n ");
  printf("\n ENO        Tensor Product Approach  (2)\n ");
  scanf("%d", &ch);
  if(ch==1) linear();
  else if(ch==2) enoTP();
}
```

Code chunk 8: ≪include≫

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#define NR_END 1
```

Benchmarks

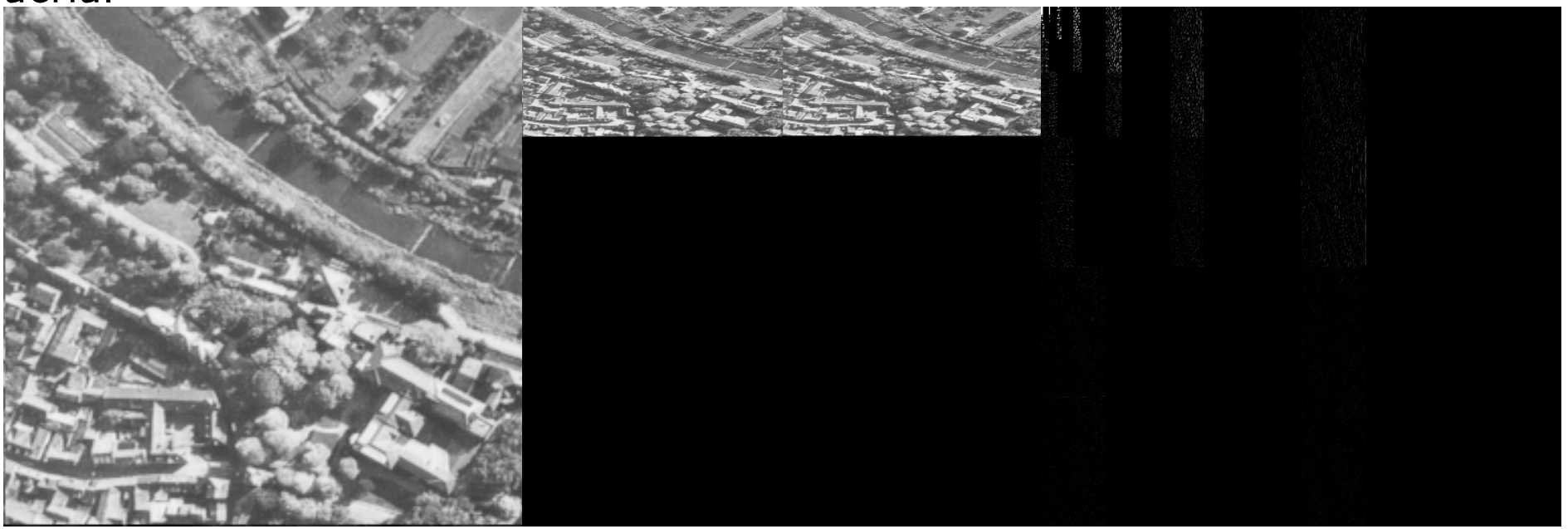
Application du programme sur une base d'exemples :

Code chunk 9: ≪benchmark≫

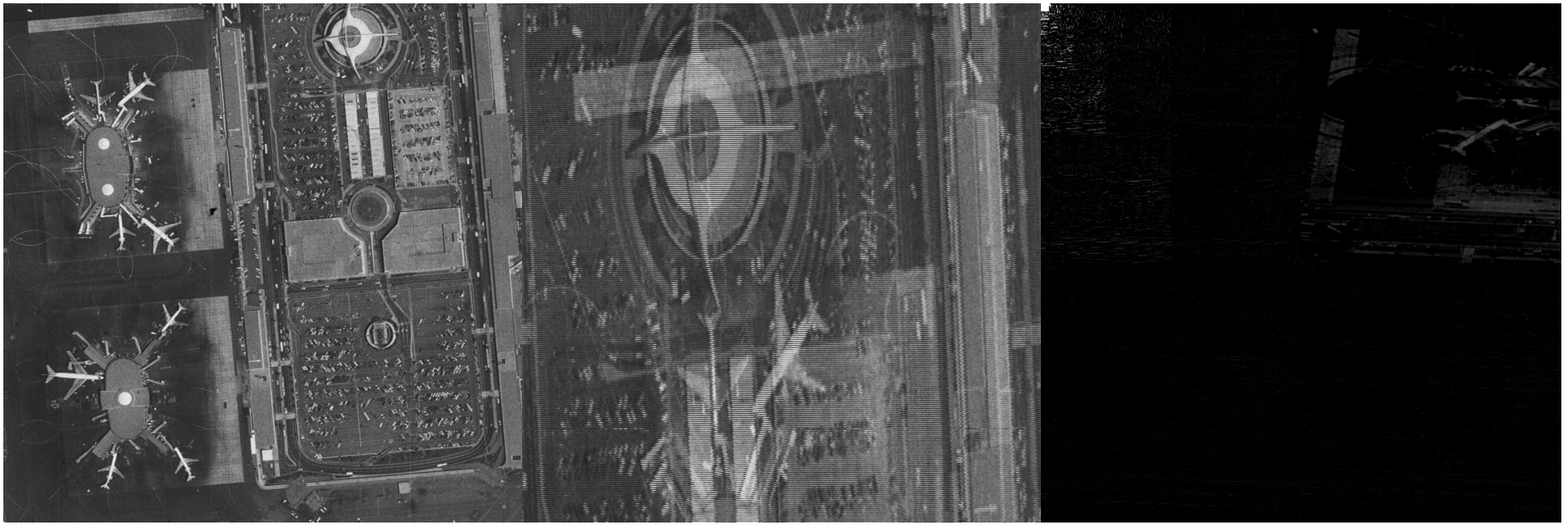
```
files=$(ls *.pgm)
for a in $files
do
  echo "\\begin"
  i=$(basename $a .pgm)
  echo "$i \\\\\\\\"
  ./linear.out $i.pgm sol_$i.pgm dec_$i.pgm 0 9 3 2 > /dev/null
  convert $i.pgm $i.pdf
  echo "\\includegraphics[width=6cm]{$i.pdf}"
  convert sol_$i.pgm sol_$i.pdf
  echo "\\includegraphics[width=6cm]{sol_$i.pdf}"
  convert dec_$i.pgm dec_$i.pdf
  echo "\\includegraphics[width=6cm]{dec_$i.pdf}\n"
done
```

aerial512

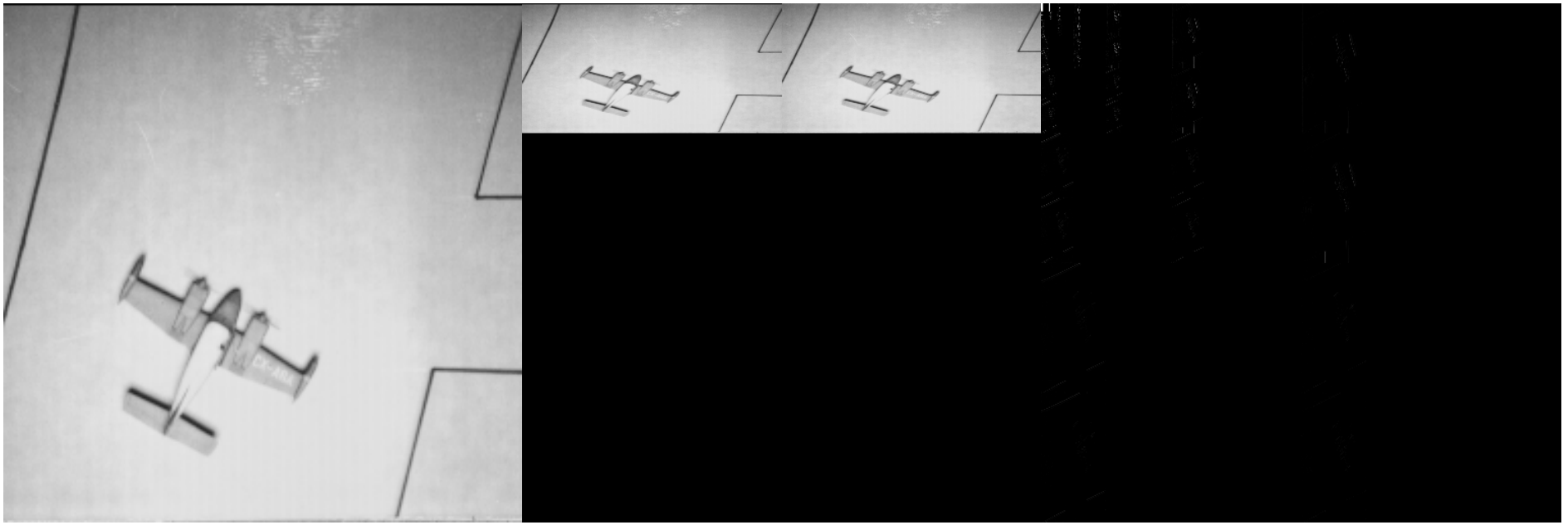
aerial



airplane1024

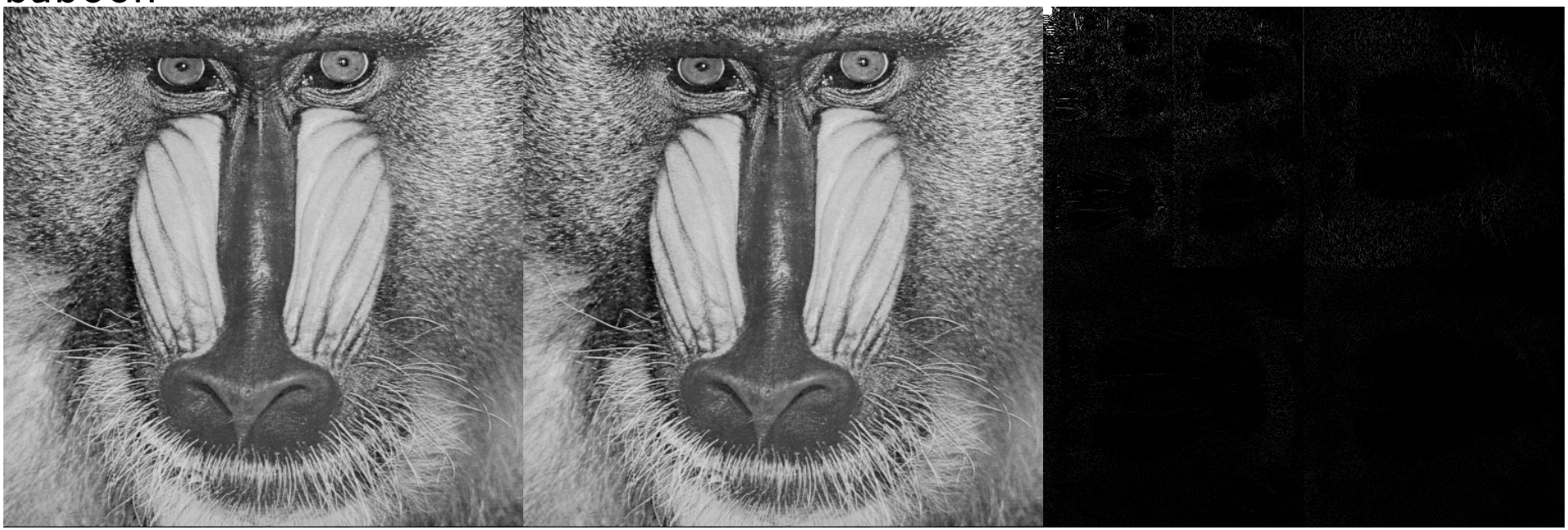


airplane256



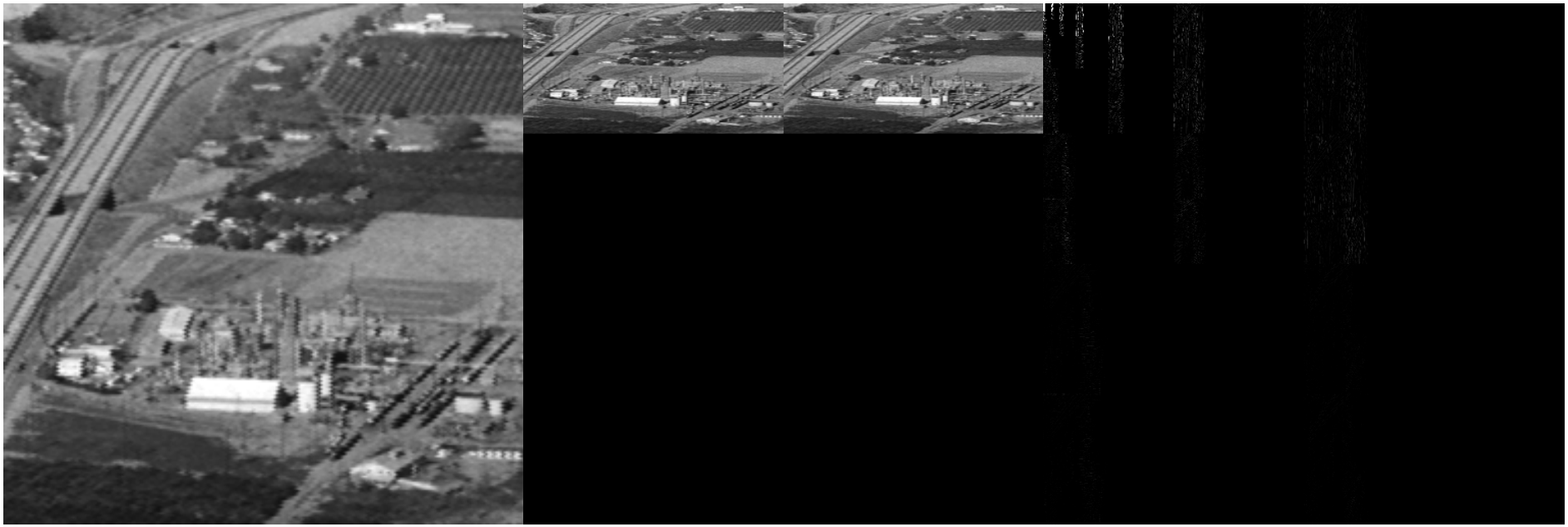
airplane-f16

baboon

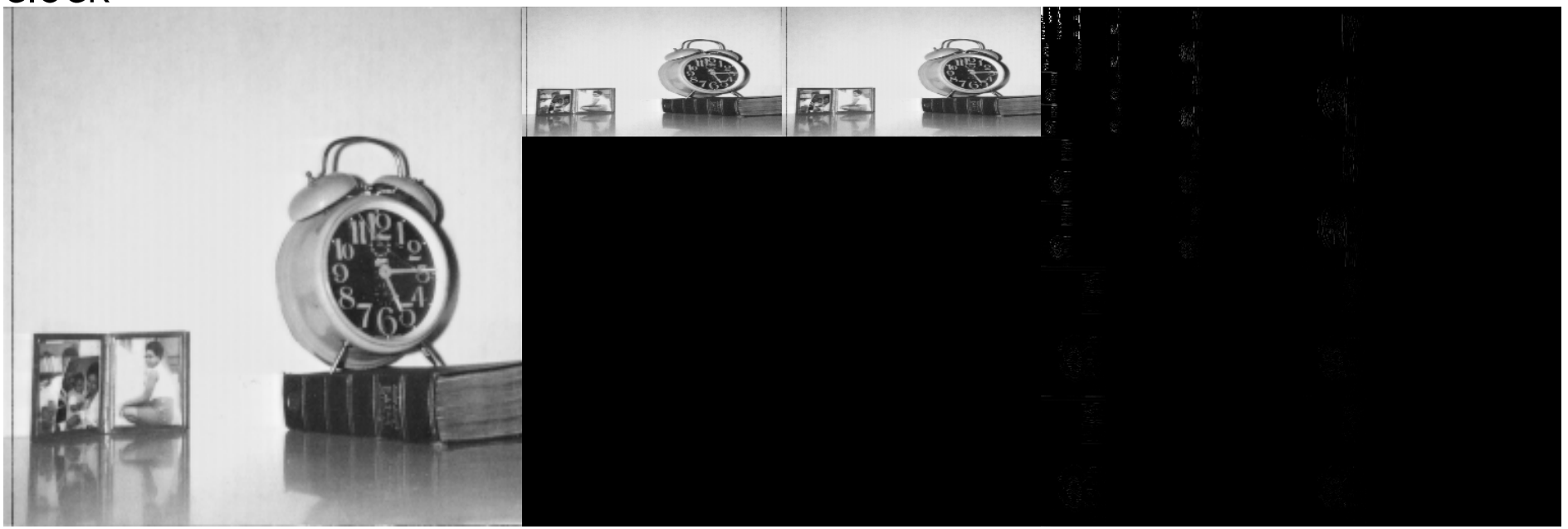


bridge

chem-plant



clock



couple512

couple

Résultat

Rapport technique complet.

Conclusion

Lepton est un outil pour la recherche computationnelle.

- produire un code source de haute qualité
  - documentation enrichie
  - code source structuré
  - réutilisation de code
  - garanties sur le code
- faire des simulations et analyser leurs résultats
  - benchmarks et bases d'exemples
  - inclusion des paramètres et des résultats
  - analyse statistique
- ré-utiliser une méthode
  - archivage
  - fournir tous les éléments pour reproduire un calcul
  - fournir l'outil pour appuyer sur entrée.

Réutilisation ?

Le rapport technique est-il utilisable par une tierce personne ?

On a tout fait pour avoir un document complet et “automatique”

- données, scripts, paramètres, faciles à modifier
- code source structuré
- documentation élaborée
- document en texte brut / LaTeX : lisible et éditable