

```

67 # CO2 expressed as a mole fraction in dry air, micromol/mol, abbreviated as ppm
68 #
69 # (-99.99 missing data; -1 no data for #daily means in month)
70 #
71 #          decimal    average    interpolated    trend    #days
72 #          date              (season corr)
73 1958    3    1958.208    315.71    315.71    314.62    -1
74 1958    4    1958.292    317.45    317.45    315.29    -1
75 1958    5    1958.375    317.50    317.50    314.71    -1
76 1958    6    1958.458    -99.99    317.10    314.85    -1
77 1958    7    1958.542    315.86    315.86    314.98    -1
78 1958    8    1958.625    314.93    314.93    315.94    -1
79 1958    9    1958.708    313.20    313.20    315.91    -1
80 1958   10    1958.792    -99.99    312.66    315.61    -1
81 1958   11    1958.875    313.33    313.33    315.31    -1
82 1958   12    1958.958    314.67    314.67    315.61    -1
83 1959    1    1959.042    315.62    315.62    315.70    -1
84 1959    2    1959.125    316.38    316.38    315.88    -1
85 1959    3    1959.208    316.71    316.71    315.62    -1

```

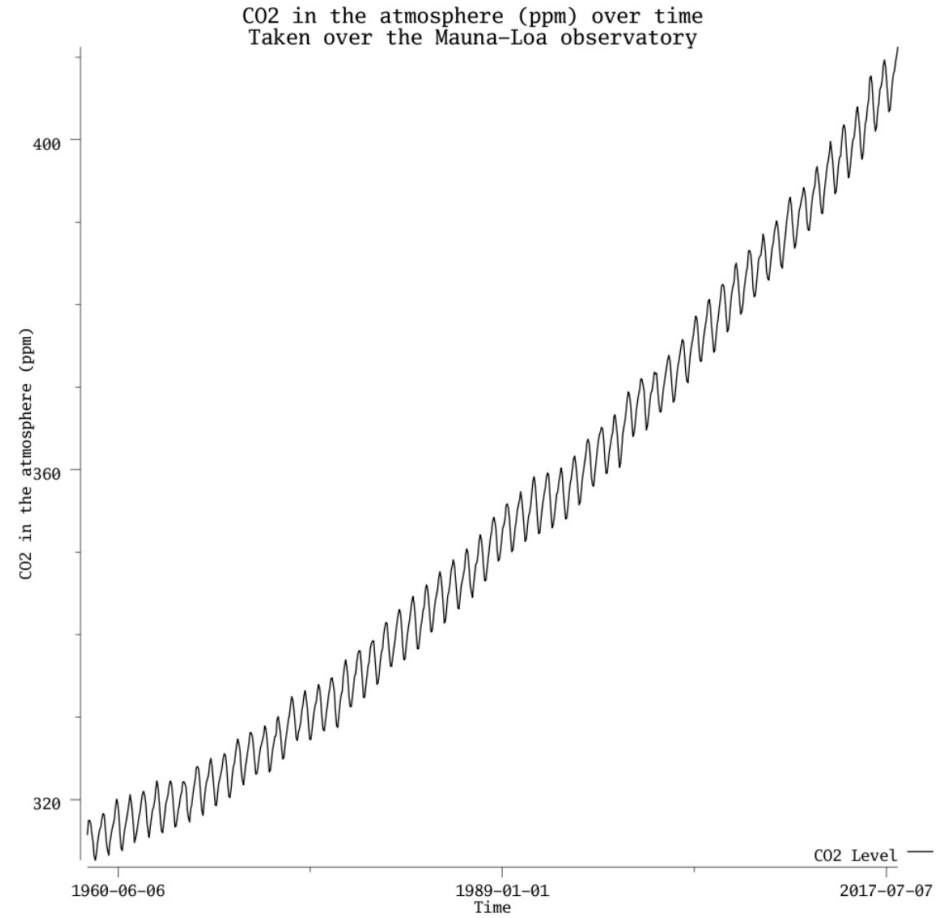
time series data values in file

main.go 1, M

Moana-Loa.png M X



Moana-Loa.png

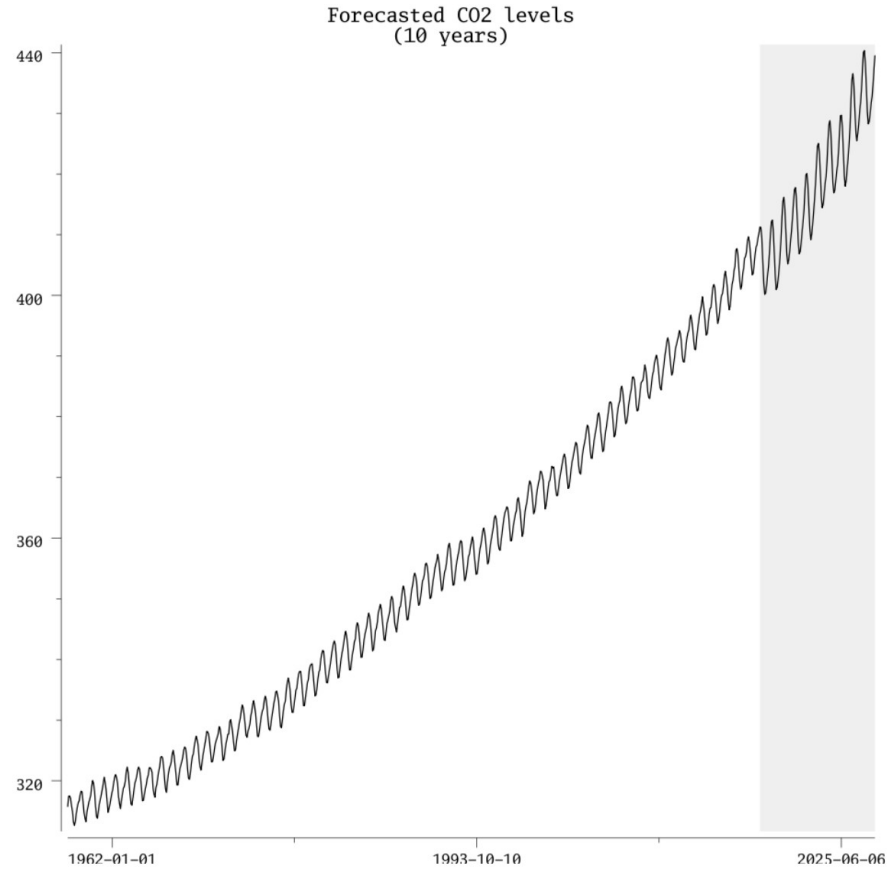


plot time series data

main.go 1, M

forecast.png M ×

forecast.png



time series data forecast

GO main.go 1, M × forecast.png M

GO main.go > main

```
54 func main() {
55     dateStrings, co2s := parse(readFromFile)
56     fmt.Printf("%T", dateStrings)
57     fmt.Printf("%T", co2s)
58     dates := parseDates(dateStrings)
59     plt := newTSPlot(dates, co2s, "CO2 Level")
60     plt.X.Label.Text = "Time"
61     plt.Y.Label.Text = "CO2 in the atmosphere (ppm)"
62     plt.Title.Text = "CO2 in the atmosphere (ppm) over time\nTaken over the Mauna-Loa observatory"
63     dieIfErr(plt.Save(25*vg.Centimeter, 25*vg.Centimeter, "Moana-Loa.png"))
64 }
```

time series data plot go code

```

main.go 1, M × forecast.png M
main.go > main
80
81 fwd := 120
82 forecast := hw(decomposed, 12, fwd, 0.1, 0.05, 0.1)
83 datesplus := forecastTime(dates, fwd)
84 forecastPlot := newTSPlot(datesplus, forecast, "")
85 maxY := math.Inf(-1)
86 minY := math.Inf(1)
87 for i := range forecast {
88     if forecast[i] > maxY {
89         maxY = forecast[i]
90     }
91     if forecast[i] < minY {
92         minY = forecast[i]
93     }
94 }
95 //extend the range a little
96 minY--
97 maxY++
98 maxX := float64(datesplus[len(datesplus)-1].Unix())
99 minX := float64(datesplus[len(dates)-1].Unix())
100
101 shadePoly := plotter.XYs{
102     {X: minX, Y: minY},
103     {X: maxX, Y: minY},
104     {X: maxX, Y: maxY},
105     {X: minX, Y: maxY},
106 }
107 poly, err := plotter.NewPolygon(shadePoly)
108 dieIfErr(err)
109 poly.Color = color.RGBA{A: 16}
110 poly.LineStyle.Color = color.RGBA{}
111 forecastPlot.Add(poly)
112
113 writeToPng(forecastPlot, "Forecasted CO2 levels\n(10 years)", "forecast.png", 25, 25)
114 }
115

```

time series data forecast go code

A text to speech using go, docker, grpc, and kubernetes,

fun > flite > ⓘ README.md > ...

```
1
2 Flite: a small run-time speech synthesis engine
3     version 2.1-release
4     Copyright Carnegie Mellon University 1999-2022
5     All rights reserved
6     http://cmuflite.org
7     https://github.com/festvox/flite
8
9
10 Flite is an open source small fast run-time text to speech engine. It
11 is the latest addition to the suite of free software synthesis tools
12 including University of Edinburgh's Festival Speech Synthesis System
13 and Carnegie Mellon University's FestVox project, tools, scripts and
14 documentation for building synthetic voices. However, flite itself
15 does not require either of these systems to compile and run.
16
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
##
#####
##
## Fast efficient small run-time speech synthesis system
## http://cmuflite.org
##
## Authors: Alan W Black (awb@cs.cmu.edu)
##          Kevin A. Lenzo (lenzo@cs.cmu.edu)
##          and others see ACKNOWLEDGEMENTS
##
## Date: Mar 2022
## Version: 2.3 current
##
#####
```

A text to speech grpc server – speech engine


```
go > src > mytts > protos > say.proto
1  syntax = "proto3";
2
3  option go_package = "myexample.com/grpc/protos";
4
5  package myexample;
6
7  service TextToSpeech {
8    |   rpc Say(Text) returns (Speech);
9  }
10
11  message Text {
12    |   string text = 1;
13  }
14
15  message Speech {
16    |   bytes Audio = 1;
17  }
18  |
```

A text to speech grpc server - proto

go > src > mytts > server >  main.go

```
26     }
27
28     s := grpc.NewServer()
29     pb.RegisterTextToSpeechServer(s, &server{})
30     log.Println("server listening at %v", lis.Addr())
31     log.Println(s)
32
33     if err = s.Serve(lis); err != nil {
34         log.Fatalf("could not serve: %v", err)
35     }
36 }
37
38 func (server) Say(ctx context.Context, in *pb.Text) (*pb.Speech, error) {
39     log.Printf("received: %v", in)
40     f, err := ioutil.TempFile("", "")
41     if err != nil {
42         return nil, fmt.Errorf("could not create tmp file: %v", err)
43     }
44     if err := f.Close(); err != nil {
45         return nil, fmt.Errorf("could not close %s: %v", f.Name(), err)
46     }
47
48     cmd := exec.Command("flite", "-t", in.Text, "-voice", "slt", "-o", f.Name())
49     if data, err := cmd.CombinedOutput(); err != nil {
50         return nil, fmt.Errorf("flite failed: %s", data)
51     }
52
53     // ... (code to read data from file and return Speech object)
54 }
```

A text to speech grpc server - server

```
go > src > mytts > client > -o main.go
```

```
7     "fmt"
8     "context"
9     "io/ioutil"
10    grpc "google.golang.org/grpc"
11
12    pb "mytts.com/grpc/protos"
13 }
14
15 func main() {
16     backend := flag.String("b", "localhost:8080", "address of the tts backend")
17     output := flag.String("o", "output.wav", "wav file saved")
18     flag.Parse()
19
20     if flag.NArg() < 1 {
21         fmt.Printf("usage:\n\t%s \\"tts\\"", os.Args[0])
22         os.Exit(1)
23     }
24     conn, err := grpc.Dial(*backend, grpc.WithInsecure())
25     if err != nil {
26         log.Fatalf("could not connect to %s: %v", *backend, err)
27     }
28     defer conn.Close()
29     client := pb.NewTextToSpeechClient(conn)
30     text := &pb.Text{Text: flag.Arg(0)}
31     res, err := client.Say(context.Background(), text)
32     if err != nil {
33         log.Fatalf("could not say %s: %v", text.Text, err )
34     }
35
36     if err := ioutil.WriteFile(*output, res.Audio, 0666); err != nil {
37         log.Fatalf("could not write to %s: %v", *output, err)
38     }
39 }
```

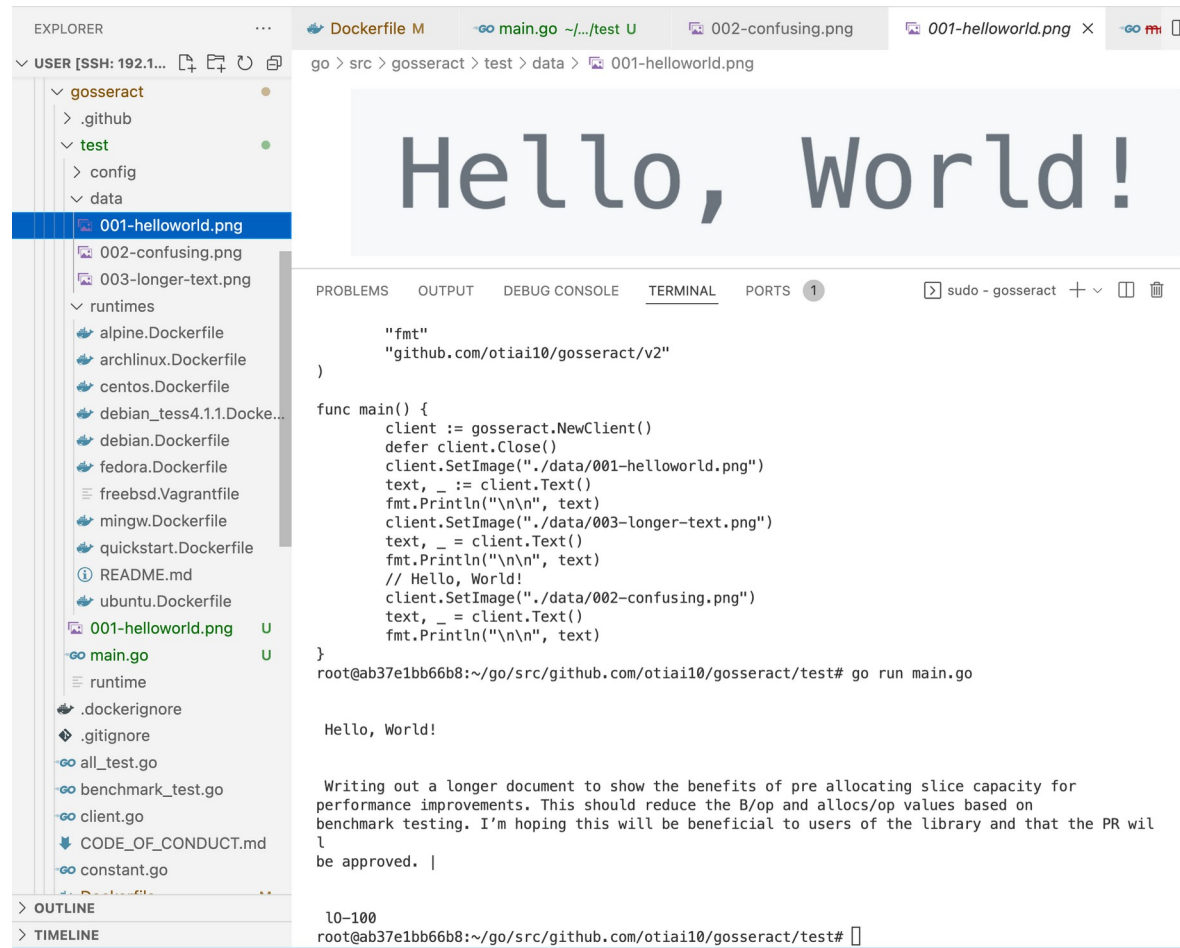
A text to speech grpc server - client

```

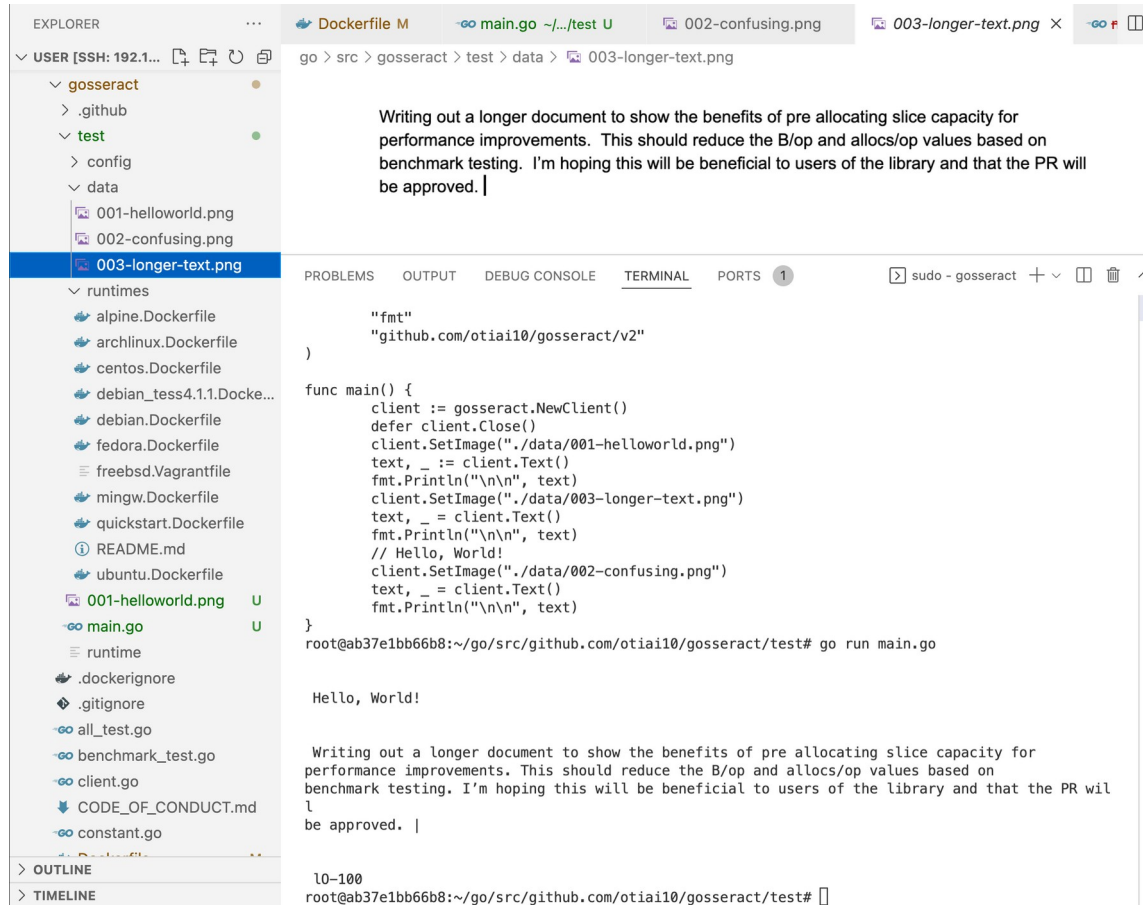
user@ml01:~/go/src/mytts/server$ sudo -E env "PATH=$PATH" make build
G005=linux go build -o app
docker build -t dev/tts .
Sending build context to Docker daemon 47.26MB
Step 1/5 : FROM ubuntu
----> 2dc39ba059dc
Step 2/5 : COPY app /app
----> 0c279ee50473
Step 3/5 : COPY flite /flite
----> 403ff0bc631f
Step 4/5 : RUN cp /flite /usr/local/bin
----> Running in 8f873ef9468c
Removing intermediate container 8f873ef9468c
----> a47b3c89e55d
Step 5/5 : ENTRYPOINT ["/app"]
----> Running in c5a80f50524e
Removing intermediate container c5a80f50524e
----> c20f0a1b6dd2
Successfully built c20f0a1b6dd2
Successfully tagged dev/tts:latest
#rm -f app
user@ml01:~/go/src/mytts/server$ sudo docker run -p 8080:8080 dev/tts --
ls
hello
2022/09/04 19:35:41 server listening at %v [::]:8080
2022/09/04 19:35:41 &{{<nil> <nil> <nil> <nil> <nil> <nil> <nil> [] [] <nil> []
0 4194304 2147483647 <nil> {0 0 0 0} {0 false} 0 0 32768 32768 120000000
000 <nil> <nil> 0} {0 0} map[] map[] false false 0xc0000d09c0 map[myexampl
e.TextToSpeech:0xc0001668a0] <nil> 0xc0000b4440 0xc0000b4460 {0 {0 0}} {{}}
0 0} 0xc000166810 0xc0000c6360 []}
2022/09/04 19:35:45 received: text:"ai summit rocks"
user@ml01:~/go/src/mytts/server$

```

A text to speech grpc server - test results



An ocr app using tesseract, go / c-binding, docker – hello world



An ocr app using tesseract, go / c-binding, docker – long text

EXPLORER

USER [SSH: 192.1...]

gosseract

- .github
- test
 - config
 - data
 - 001-helloworld.png
 - 002-confusing.png
 - 003-longer-text.png
 - runtimes
 - alpine.Dockerfile
 - archlinux.Dockerfile
 - centos.Dockerfile
 - debian_tess4.1.1.Docke...
 - debian.Dockerfile
 - fedora.Dockerfile
 - freebsd.Vagrantfile
 - mingw.Dockerfile
 - quickstart.Dockerfile
 - README.md
 - ubuntu.Dockerfile
 - 001-helloworld.png U
 - main.go U
 - runtime
 - .dockerignore
 - .gitignore
 - all_test.go
 - benchmark_test.go
 - client.go
 - CODE_OF_CONDUCT.md
 - constant.go

OUTLINE

TIMELINE

Dockerfile M

main.go ~/.../test U

002-confusing.png x

001-helloworld.png

go > src > gosseract > test > data > 002-confusing.png

IO-100

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 1

sudo - gosseract + -

```
"fmt"
"github.com/otiai10/gosseract/v2"
)

func main() {
    client := gosseract.NewClient()
    defer client.Close()
    client.SetImage("./data/001-helloworld.png")
    text, _ := client.Text()
    fmt.Println("\n\n", text)
    client.SetImage("./data/003-longer-text.png")
    text, _ = client.Text()
    fmt.Println("\n\n", text)
    // Hello, World!
    client.SetImage("./data/002-confusing.png")
    text, _ = client.Text()
    fmt.Println("\n\n", text)
}

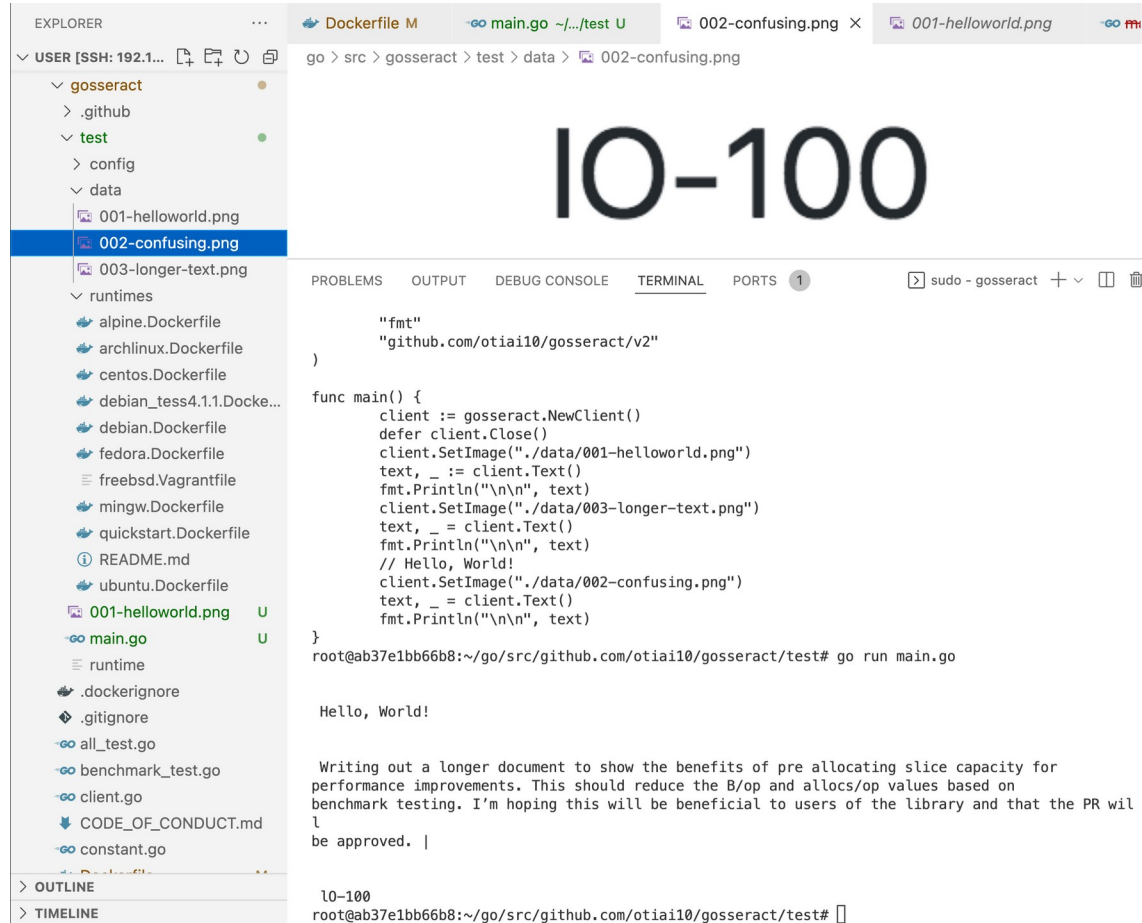
root@ab37e1bb66b8:~/go/src/github.com/otiai10/gosseract/test# go run main.go

Hello, World!

Writing out a longer document to show the benefits of pre allocating slice capacity for
performance improvements. This should reduce the B/op and allocs/op values based on
benchmark testing. I'm hoping this will be beneficial to users of the library and that the PR wil
l
be approved. |

10-100
root@ab37e1bb66b8:~/go/src/github.com/otiai10/gosseract/test#
```

An ocr app using tesseract, go / c-binding, docker – confusing text



An ocr app using tesseract, go / c-binding, docker – confusing text

The image shows a VS Code editor window with a file explorer on the left and a terminal on the right. The file explorer shows a project structure with a 'src' directory containing 'goml', 'gonum-plot', 'gorgonia', 'gosseract', 'grpc-go', 'mytts', and an 'ocrserver' directory. The 'ocrserver' directory contains '.github', 'app', 'controllers', 'filters', 'test', '.gitignore', 'docker-compose.yml', 'Dockerfile', 'Dockerfile.bak', 'go.mod', 'go.sum', 'heroku.yml', 'LICENSE', 'main.go', 'README.md', 'tensorflow-go', 'tts.old', 'webfileserver', 'Music', 'Pictures', 'Public', 'snap', and 'OUTLINE'. The terminal shows the Dockerfile content and the output of the build process.

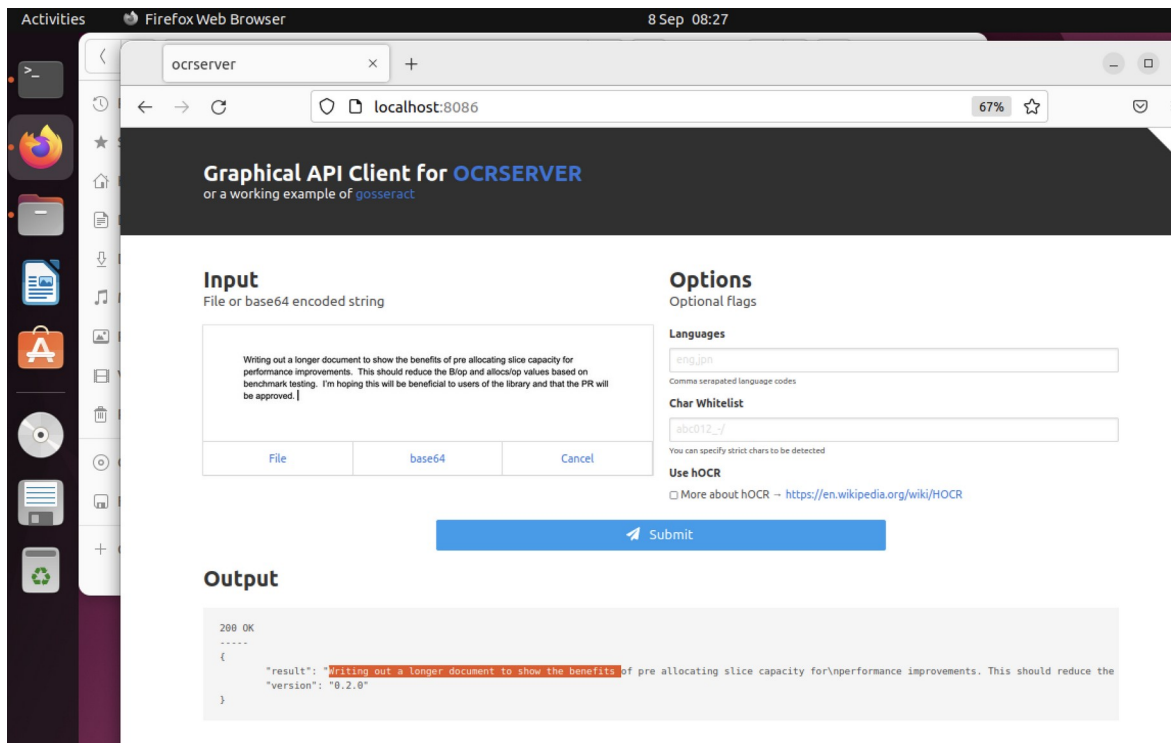
```
go > src > ocrserver > Dockerfile
13  ENV G011MODULE=on
14  ENV GOPATH=${HOME}/go
15  ENV PATH=${PATH}:${GOPATH}/bin
16
17  ADD . $GOPATH/src/github.com/otiai10/ocrserver
18  WORKDIR $GOPATH/src/github.com/otiai10/ocrserver
19  RUN go get -v ./... && go install .
20
21  # Load languages
22  RUN if [ -n "${LOAD_LANG}" ]; then apt-get install -y tesseract-ocr-${LOAD_LANG}; fi
23
24  ENV PORT=8086
25  CMD ["ocrserver"]
26
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
sudo - ocrserver + - - - - -
(Reading database ... 26976 files and directories currently installed.)
Preparing to unpack .../tesseract-ocr-jpn_1%3a4.00~git30-7274cfa-1.1_all.deb ...
Unpacking tesseract-ocr-jpn (1:4.00~git30-7274cfa-1.1) ...
Setting up tesseract-ocr-jpn (1:4.00~git30-7274cfa-1.1) ...
Removing intermediate container c5a2f558bea9
--> 0caf493aa73b
Step 12/13 : ENV PORT=8086
--> Running in 2dd1db0fab39
Removing intermediate container 2dd1db0fab39
--> c2e9e4f86a96
Step 13/13 : CMD ["ocrserver"]
--> Running in 8e389a00b3c6
Removing intermediate container 8e389a00b3c6
--> 9bd03430ca78
Successfully built 9bd03430ca78
Successfully tagged dev/ocrs:latest
user@ml01:~/go/src/ocrserver$ sudo docker run -p 8086:8086 dev/ocrs
[ocrserver] listening on port 8086

[ocrserver] GET /
[ocrserver] POST /file
[ocrserver] POST /file
[ocrserver] POST /file
```

An ocr app using tesseract, go / c-binding, docker – build / run docker server



An ocr app using tesseract, go / c-binding, docker – docker ocr server gui and results

```

user@ml01:~/go/src/mytts/server$ sudo -E env "PATH=$PATH" make build
G005=linux go build -o app
docker build -t dev/tts .
Sending build context to Docker daemon 47.26MB
Step 1/5 : FROM ubuntu
----> 2dc39ba059dc
Step 2/5 : COPY app /app
----> 0c279ee50473
Step 3/5 : COPY flite /flite
----> 403ff0bc631f
Step 4/5 : RUN cp /flite /usr/local/bin
----> Running in 8f873ef9468c
Removing intermediate container 8f873ef9468c
----> a47b3c89e55d
Step 5/5 : ENTRYPOINT ["/app"]
----> Running in c5a80f50524e
Removing intermediate container c5a80f50524e
----> c20f0a1b6dd2
Successfully built c20f0a1b6dd2
Successfully tagged dev/tts:latest
#rm -f app
user@ml01:~/go/src/mytts/server$ sudo docker run -p 8080:8080 dev/tts --
ls
hello
2022/09/04 19:35:41 server listening at %v [::]:8080
2022/09/04 19:35:41 &{{<nil> <nil> <nil> <nil> <nil> <nil> [] [] <nil> []
0 4194304 2147483647 <nil> {0 0 0 0} {0 false} 0 0 32768 32768 120000000
000 <nil> <nil> 0} {0 0} map[] map[] false false 0xc000d09c0 map[myexampl
e.TextToSpeech:0xc0001668a0] <nil> 0xc0000b4440 0xc0000b4460 {0 {0 0}} {{}
0 0} 0xc000166810 0xc0000c6360 []}
2022/09/04 19:35:45 received: text:"ai summit rocks"
user@ml01:~/go/src/mytts/server$

```

```

user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
rocks"
2022/09/04 20:32:
vailable desc = c
l tcp 127.0.0.1:8
exit status 1
user@ml01:~/go/src
rocks"
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
rocks"
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
rocks"
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src

```

A text to speech grpc server - test results

```

user@ml01:~/go/src/mytts/server$ sudo -E env "PATH=$PATH" make build
G005=linux go build -o app
docker build -t dev/tts .
Sending build context to Docker daemon 47.26MB
Step 1/5 : FROM ubuntu
----> 2dc39ba059dc
Step 2/5 : COPY app /app
----> 0c279ee50473
Step 3/5 : COPY flite /flite
----> 403ff0bc631f
Step 4/5 : RUN cp /flite /usr/local/bin
----> Running in 8f873ef9468c
Removing intermediate container 8f873ef9468c
----> a47b3c89e55d
Step 5/5 : ENTRYPOINT ["/app"]
----> Running in c5a80f50524e
Removing intermediate container c5a80f50524e
----> c20f0a1b6dd2
Successfully built c20f0a1b6dd2
Successfully tagged dev/tts:latest
#rm -f app
user@ml01:~/go/src/mytts/server$ sudo docker run -p 8080:8080 dev/tts --
ls
hello
2022/09/04 19:35:41 server listening at %v [::]:8080
2022/09/04 19:35:41 &{{<nil> <nil> <nil> <nil> <nil> <nil> [] [] <nil> []
0 4194304 2147483647 <nil> {0 0 0 0} {0 false} 0 0 32768 32768 120000000
000 <nil> <nil> 0} {0 0} map[] map[] false false 0xc000d09c0 map[myexampl
e.TextToSpeech:0xc0001668a0] <nil> 0xc0000b4440 0xc0000b4460 {0 {0 0}} {{}
0 0} 0xc000166810 0xc0000c6360 []}
2022/09/04 19:35:45 received: text:"ai summit rocks"
user@ml01:~/go/src/mytts/server$

```

```

user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
rocks"
2022/09/04 20:32:
vailable desc = c
l tcp 127.0.0.1:8
exit status 1
user@ml01:~/go/src
rocks"
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
rocks"
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
rocks"
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src
Playing WAVE 'out
user@ml01:~/go/src

```

A text to speech grpc server - test results