# mandulia

Claude Heiland-Allen

2010–2018

# Contents

# 1 AUTHORS

```
Claude Heiland-Allen
  claude@mathr.co.uk
  http://mathr.co.uk
```

# 2 config/defaults.lua

```
--[[--

Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.

--]]--

do

  local m = mandulia -- shorter aliased name for clarity

  -- print(unpack(m.args))    -- extra command line arguments are here

  -- items marked (*) take effect only at initialization time
  -- unmarked items can be changed at any time
  m.record      = false       --     dump frames as PPM to stdout
  m.fullscreen  = false       --     full screen mode
  m.width       = 1024        -- (*) window width
  m.height      =  576        -- (*) window height
  m.fps         =   25        --     frames per second
  m.detail      =   11        -- (*) detail level
  if m.width > m.height then  --     display size of each Julia set
    m.displaysize = m.width / 16
  else
    m.displaysize = m.height / 16
  end
  m.juliasize   =  256        -- (*) pixel size of each Julia set
  m.jobs        =  512        -- (*) maximum number of jobs per frame
  m.images      =  512        -- (*) maximum number of images to cache
  m.textures    = 2048        -- (*) maximum number of textures to cache
  m.workers     =    2        -- (*) number of worker threads
  m.view        = { x = 0, y = 0, z = 0 } -- viewport coordinates

  -- gets called when the window size changes
  function m.reshape(width, height)
    m.width  = width
    m.height = height
    if width > height then
      m.displaysize = width / 16
    else
      m.displaysize = height / 16
    end
  end

  -- gets called at the start of each frame
  -- eg: set 'view' here for animations
  function m.render()
  end

  -- gets called when a key is pressed
  -- eg: call 'quit()' if you want to quit
  function m.keyboard(key)
  end

  -- gets called when the program is exiting
  function m.atexit()
  end

end
```

# 3 config/distance.lua

```
--[[--

Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.

--]]--

do

  --[[--

  SYNOPSIS

    s = exteriordistance(x, y, n, r)

  INPUTS

    x, y  :  point in the complex plane
    n      :  iteration limit
    r      :  escape radius

  OUTPUTS

    s      :  distance from Mandelbrot set (or nil if inside)

  EXAMPLE

    s = exteriordistance(x, y, 100, 100)
    if s ~= nil and s < 0.01 then ... else ... end

  --]]--
  function exteriordistance(x, y, n, r)

    local px = 0
    local py = 0
    local cx = x
    local cy = y
    local dx = 1
    local dy = 0
    local r2 = r * r
    local escape = false
    local px1, py1, dx1, dy1, px2, py2, pxy, d2, p, d

    for i = 1,n do
      -- p_n+1 := p_n * p_n + c
      -- d_n+1 := 2 * p_n * d_n + 1
      px2 = px * px
      py2 = py * py
      d2 = px2 + py2
      if d2 > r2 then escape = true ; break end
      pxy = px * py
      px1 = px2 - py2 + cx
      py1 = 2 * pxy + cy
      dx1 = 2 * (px * dx - py * dy) + 1
      dy1 = 2 * (dx * py + dy * px)
      px = px1 ; py = py1 ; dx = dx1 ; dy = dy1
    end

    if escape then
      p = math.sqrt(px * px + py * py)
      d = math.sqrt(dx * dx + dy * dy)
      return 2 * p * math.log(p) / d
    else
      return nil
    end

  end

end
```

## 4 config/main.lua

```
--[[--

Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.

--]]--

require("defaults")
require("transition")
require("distance")

do

  -- frame count without interaction
  local mode = "interact"
  local unattended = 0
  local attentionSpan = 25 * 60

  -- frame counts for recording
  local frames = 0
  local frameLimit = 0

  -- magic numbers
  math.randomseed(os.time())
  local phi  = (math.sqrt(5) + 1) / 2
  local phi1 = (math.sqrt(5) - 1) / 2

  -- animation parameters
  local defaultSpeed = 0.125
  local defaultWeight = 8
  local speed = defaultSpeed
  local weight = defaultWeight

  -- view bounds
  local minimum = { x = -4, y = -4, z =  0 }
  local maximum = { x =  4, y =  4, z = 42 }
  local function clamp(x, mi, ma)
    return math.min(math.max(x, mi), ma)
  end

  -- attract mode for interesting behaviour when unattended
  local attract = { }
  do

    local function walk(x, mi, ma, d)
      local p = (x - mi) / (ma - mi)
      local s = math.random()
      return x + d * (s - p)
    end

    -- default view
    local function zero() return { x = 0, y = 0, z = minimum.z } end

    -- animation variables
    local source = zero()
    local target = zero()
    local t = 0
    local s = 0
    local f = function(t) return zero() end
    local dz = minimum.z

    local history = { }
    for i = 1,48 do
      history[i] = { x = 0, y = 0 }
    end
    local historyptr = 1

    attract.enter = function()
      mode = "attract"
      target.x = mandulia.view.x
      target.y = mandulia.view.y
      target.z = mandulia.view.z
      t = 0
      s = 0
    end

    attract.render = function()
      t = t + speed * (phi1 ^ dz)
      if t > s then -- we reached the target, new target needed
        t = t - s
        source.x = target.x
        source.y = target.y
        source.z = target.z
        local interesting
        local x, y, e, v, d2, dx, dy
        local r = 25 * phi1 ^ dz
        e = exteriordistance(source.x, source.y, 1000, 1000)
        interesting = e ~= nil and e < 0.00001
        if interesting then
          interesting = false
          while not interesting do
            x = source.x + r * (2 * math.random() - 1)
            y = source.y + r * (2 * math.random() - 1)
            e = exteriordistance(x, y, 1000, 1000)
            if e ~= nil and e < 0.00001 then
              v = false
              for i,p in ipairs(history) do
                dx = x - p.x
                dy = y - p.y
                d2 = dx * dx + dy * dy
                v = v or d2 < 0.01
              end
              if v then r = r * 1.1 end
              interesting = not v
```

```lua
            end
          end
        else
          interesting = false
          while not interesting do
            x = math.random() * (maximum.x - minimum.x) + minimum.x
            y = math.random() * (maximum.y - minimum.y) + minimum.y
            e = exteriordistance(x, y, 1000, 1000)
            interesting = e ~= nil and e < 0.00001
          end
        end
        target.x = clamp(x, minimum.x, maximum.x)
        target.y = clamp(y, minimum.y, maximum.y)
        target.z = walk(source.z, minimum.z, maximum.z, 3)
        history[historyptr].x = target.x
        history[historyptr].y = target.y
        historyptr = historyptr + 1
        if historyptr > #history then historyptr = 1 end
        s, f = transition(source, target, weight, phi)
      end
      mandulia.view = f(t)
      mandulia.view.z = clamp(mandulia.view.z, minimum.z, maximum.z)
      dz = mandulia.view.z
    end -- render

  end -- attract

  -- interact mode for engagement and control
  local interact = { }
  do

    local delta = { x = 0, y = 0, z = 0 }

    interact.enter = function()
      mode = "interact"
      unattended = 0
    end

    interact.render = function()
      local x = mandulia.view.x + speed * delta.x * phi1 ^ mandulia.view.z
      local y = mandulia.view.y + speed * delta.y * phi1 ^ mandulia.view.z
      local z = mandulia.view.z + speed * delta.z
      mandulia.view.x = clamp(x, minimum.x, maximum.x)
      mandulia.view.y = clamp(y, minimum.y, maximum.y)
      mandulia.view.z = clamp(z, minimum.z, maximum.z)
    end

    local keys =
      { Right    = function() delta.x = delta.x + 1 end
      , Left     = function() delta.x = delta.x - 1 end
      , Up       = function() delta.y = delta.y + 1 end
      , Down     = function() delta.y = delta.y - 1 end
      , PageUp   = function() delta.z = delta.z + 1 end
      , PageDown = function() delta.z = delta.z - 1 end
      , End      = function() delta         = { x = 0, y = 0, z = 0 } end
      , Home     = function() delta         = { x = 0, y = 0, z = 0 }
                              mandulia.view = { x = 0, y = 0, z = 0 } end
      , ["["]    = function() speed = speed * 0.95 end
      , ["]"]    = function() speed = speed / 0.95 end
      , ["#"]    = function() speed = defaultSpeed end
      , ["{"]    = function() weight = weight * 0.95 end
      , ["}"]    = function() weight = weight / 0.95 end
      , ["~"]    = function() weight = defaultWeight end
      }

    interact.keyboard = function(key)
      if type(keys[key]) == "function" then keys[key]() end
    end

  end -- interact

  local screenshot = { }
  local record = { }
  do

    local recording = false

    screenshot.enter = function()
      frames = 0
      frameLimit = 1
      frameStep = 1
      recording = true
    end

    record.enter = function()
      frames = 0
      frameLimit = 25 * 60 * 10
      frameStep = 1
      recording = true
    end

    record.render = function()
      if recording then
        mandulia.record = (frames % frameStep) == 0 and frames < frameLimit
        frames = frames + 1
      end
    end

  end -- screenshot/record


  function mandulia.render()
    if mode == "interact" then
      interact.render()
      unattended = unattended + 1
      if unattended >= attentionSpan then attract.enter() end
    elseif mode == "attract" then
      attract.render()
    end
    record.render()
  end

  function mandulia.keyboard(key)
    if     key == "Escape" then mandulia.quit()
```

```
    elseif key == "F11"    then mandulia.fullscreen = not mandulia.fullscreen
    elseif key == "a"      then attract.enter()
    elseif key == "r"      then record.enter()
    elseif key == "s"      then screenshot.enter()
    else
      interact.enter()
      interact.keyboard(key)
    end
  end

end

pcall(function() require("config") end)
```

## 5 config/transition.lua

```
--[[--

Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.

--]]--

do

  --[[--

  SYNOPSIS

    s, f = transition(a, b, p, q)

  INPUTS

    a ~ {x,y,z}  z >= 0      :  source viewpoint coordinates
    b ~ {x,y,z}  z >= 0      :  target viewpoint coordinates
    p > 0                    :  zoom vs move weighting (smaller => zoomier)
    q > 1                    :  zoom factor

  OUTPUTS

    s >= 0                   :  source->target path length
    f(t) ~ {x,y,z}  z >= 0   :  viewpoint interpolator
      where 0 <= t <= s
  EXAMPLE

    s, f = transition(a, b, p, q)
    for t = 0,s,dt do
      view = f(t)
    end

  --]]--
  function transition(aa, bb, pp, qq)

    -- copy arguments
    local a = { x = aa.x, y = aa.y, z = aa.z }
    local b = { x = bb.x, y = bb.y, z = bb.z }
    local p = pp
    local q = qq

    -- transform coordinates
    local logp = math.log(p)
    local logq = math.log(q)
    local function y(z) return p * (q ^ (-z)) end
    local function z(y)
      if y > 0 then
        return math.max((logp - math.log(y)) / logq, 0)
      else
        return 0
      end
    end

    -- initial coordinates
    local dx = b.x - a.x
    local dy = b.y - a.y
    -- local x0 = 0
    local y0 = y(a.z)
    local x1 = math.sqrt(dx * dx + dy * dy)
    local y1 = y(b.z)

    if x1 > 0 then    -- circular arc centered on x-axis

      local xc = (x1*x1 + y1*y1 - y0*y0) / (2 * x1)
      local a0 = math.atan2(y0,    - xc)
      local a1 = math.atan2(y1, x1 - xc)
      local r  = math.sqrt(xc*xc + y0*y0)
      local s  = r * math.abs(a1 - a0)
      local da = (a1 - a0) / s
      local fx = dx / x1
      local fy = dy / x1
      return s, function(t)
        local at = a0 + t * da
        local dr = xc + r * math.cos(at)
        return { x = a.x + fx * dr
               , y = a.y + fy * dr
               , z = z(r * math.sin(at))
               }
      end

    else
      local s = math.abs(y1 - y0)
```

```
      if s > 0 then  -- vertical line segment

        local ds = (y1 - y0) / s
        return s, function(t)
          return { x = a.x, y = a.y, z = z(y0 + t * ds) }
        end

      else           -- end points are identical

        return 0, function(t)
          return { x = a.x, y = a.y, z = a.z }
        end

      end
    end
  end

end
```

# 6 COPYING

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The GNU General Public License is a free, copyleft license for
software and other kinds of works.

  The licenses for most software and other practical works are designed
to take away your freedom to share and change the works.  By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.  We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors.  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

  To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights.  Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received.  You must make sure that they, too, receive
```

```
or can get the source code.  And you must show them these terms so they
know their rights.

  Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

  For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software.  For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

  Some devices are designed to deny users access to install or run
modified versions of the software inside them, although the manufacturer
can do so.  This is fundamentally incompatible with the aim of
protecting users' freedom to change the software.  The systematic
pattern of such abuse occurs in the area of products for individuals to
use, which is precisely where it is most unacceptable.  Therefore, we
have designed this version of the GPL to prohibit the practice for those
products.  If such problems arise substantially in other domains, we
stand ready to extend this provision to those domains in future versions
of the GPL, as needed to protect the freedom of users.

  Finally, every program is threatened constantly by software patents.
States should not allow patents to restrict development and use of
software on general-purpose computers, but in those that do, we wish to
avoid the special danger that patents applied to a free program could
make it effectively proprietary.  To prevent this, the GPL assures that
patents cannot be used to render the program non-free.

  The precise terms and conditions for copying, distribution and
modification follow.

                       TERMS AND CONDITIONS

  0. Definitions.

  ”This License” refers to version 3 of the GNU General Public License.

  ”Copyright” also means copyright-like laws that apply to other kinds of
works, such as semiconductor masks.

  ”The Program” refers to any copyrightable work licensed under this
License.  Each licensee is addressed as ”you”.  ”Licensees” and
”recipients” may be individuals or organizations.

  To ”modify” a work means to copy from or adapt all or part of the work
in a fashion requiring copyright permission, other than the making of an
exact copy.  The resulting work is called a ”modified version” of the
earlier work or a work ”based on” the earlier work.

  A ”covered work” means either the unmodified Program or a work based
on the Program.

  To ”propagate” a work means to do anything with it that, without
permission, would make you directly or secondarily liable for
```

infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To ”convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays ”Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The ”source code” for a work means the preferred form of the work for making modifications to it. ”Object code” means any non-source form of a work.

A ”Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The ”System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A ”Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The ”Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to ”keep intact all notices”.

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an ”aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no

```
    more than your reasonable cost of physically performing this
    conveying of source, or (2) access to copy the
    Corresponding Source from a network server at no charge.

    c) Convey individual copies of the object code with a copy of the
    written offer to provide the Corresponding Source.  This
    alternative is allowed only occasionally and noncommercially, and
    only if you received the object code with such an offer, in accord
    with subsection 6b.

    d) Convey the object code by offering access from a designated
    place (gratis or for a charge), and offer equivalent access to the
    Corresponding Source in the same way through the same place at no
    further charge.  You need not require recipients to copy the
    Corresponding Source along with the object code.  If the place to
    copy the object code is a network server, the Corresponding Source
    may be on a different server (operated by you or a third party)
    that supports equivalent copying facilities, provided you maintain
    clear directions next to the object code saying where to find the
    Corresponding Source.  Regardless of what server hosts the
    Corresponding Source, you remain obligated to ensure that it is
    available for as long as needed to satisfy these requirements.

    e) Convey the object code using peer-to-peer transmission, provided
    you inform other peers where the object code and Corresponding
    Source of the work are being offered to the general public at no
    charge under subsection 6d.

  A separable portion of the object code, whose source code is excluded
from the Corresponding Source as a System Library, need not be
included in conveying the object code work.

  A ”User Product” is either (1) a ”consumer product”, which means any
tangible personal property which is normally used for personal, family,
or household purposes, or (2) anything designed or sold for incorporation
into a dwelling.  In determining whether a product is a consumer product,
doubtful cases shall be resolved in favor of coverage.  For a particular
product received by a particular user, ”normally used” refers to a
typical or common use of that class of product, regardless of the status
of the particular user or of the way in which the particular user
actually uses, or expects or is expected to use, the product.  A product
is a consumer product regardless of whether the product has substantial
commercial, industrial or non-consumer uses, unless such uses represent
the only significant mode of use of the product.

  ”Installation Information” for a User Product means any methods,
procedures, authorization keys, or other information required to install
and execute modified versions of a covered work in that User Product from
a modified version of its Corresponding Source.  The information must
suffice to ensure that the continued functioning of the modified object
code is in no case prevented or interfered with solely because
modification has been made.

  If you convey an object code work under this section in, or with, or
specifically for use in, a User Product, and the conveying occurs as
part of a transaction in which the right of possession and use of the
User Product is transferred to the recipient in perpetuity or for a
```

```
fixed term (regardless of how the transaction is characterized), the
Corresponding Source conveyed under this section must be accompanied
by the Installation Information.  But this requirement does not apply
if neither you nor any third party retains the ability to install
modified object code on the User Product (for example, the work has
been installed in ROM).

  The requirement to provide Installation Information does not include a
requirement to continue to provide support service, warranty, or updates
for a work that has been modified or installed by the recipient, or for
the User Product in which it has been modified or installed.  Access to a
network may be denied when the modification itself materially and
adversely affects the operation of the network or violates the rules and
protocols for communication across the network.

  Corresponding Source conveyed, and Installation Information provided,
in accord with this section must be in a format that is publicly
documented (and with an implementation available to the public in
source code form), and must require no special password or key for
unpacking, reading or copying.

  7. Additional Terms.

  ”Additional permissions” are terms that supplement the terms of this
License by making exceptions from one or more of its conditions.
Additional permissions that are applicable to the entire Program shall
be treated as though they were included in this License, to the extent
that they are valid under applicable law.  If additional permissions
apply only to part of the Program, that part may be used separately
under those permissions, but the entire Program remains governed by
this License without regard to the additional permissions.

  When you convey a copy of a covered work, you may at your option
remove any additional permissions from that copy, or from any part of
it.  (Additional permissions may be written to require their own
removal in certain cases when you modify the work.)  You may place
additional permissions on material, added by you to a covered work,
for which you have or can give appropriate copyright permission.

  Notwithstanding any other provision of this License, for material you
add to a covered work, you may (if authorized by the copyright holders of
that material) supplement the terms of this License with terms:

    a) Disclaiming warranty or limiting liability differently from the
    terms of sections 15 and 16 of this License; or

    b) Requiring preservation of specified reasonable legal notices or
    author attributions in that material or in the Appropriate Legal
    Notices displayed by works containing it; or

    c) Prohibiting misrepresentation of the origin of that material, or
    requiring that modified versions of such material be marked in
    reasonable ways as different from the original version; or

    d) Limiting the use for publicity purposes of names of licensors or
    authors of the material; or
```

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered ”further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

```
  You are not required to accept this License in order to receive or
run a copy of the Program.  Ancillary propagation of a covered work
occurring solely as a consequence of using peer-to-peer transmission
to receive a copy likewise does not require acceptance.  However,
nothing other than this License grants you permission to propagate or
modify any covered work.  These actions infringe copyright if you do
not accept this License.  Therefore, by modifying or propagating a
covered work, you indicate your acceptance of this License to do so.

  10. Automatic Licensing of Downstream Recipients.

  Each time you convey a covered work, the recipient automatically
receives a license from the original licensors, to run, modify and
propagate that work, subject to this License.  You are not responsible
for enforcing compliance by third parties with this License.

  An "entity transaction" is a transaction transferring control of an
organization, or substantially all assets of one, or subdividing an
organization, or merging organizations.  If propagation of a covered
work results from an entity transaction, each party to that
transaction who receives a copy of the work also receives whatever
licenses to the work the party's predecessor in interest had or could
give under the previous paragraph, plus a right to possession of the
Corresponding Source of the work from the predecessor in interest, if
the predecessor has it or can get it with reasonable efforts.

  You may not impose any further restrictions on the exercise of the
rights granted or affirmed under this License.  For example, you may
not impose a license fee, royalty, or other charge for exercise of
rights granted under this License, and you may not initiate litigation
(including a cross-claim or counterclaim in a lawsuit) alleging that
any patent claim is infringed by making, using, selling, offering for
sale, or importing the Program or any portion of it.

  11. Patents.

  A "contributor" is a copyright holder who authorizes use under this
License of the Program or a work on which the Program is based.  The
work thus licensed is called the contributor's "contributor version".

  A contributor's "essential patent claims" are all patent claims
owned or controlled by the contributor, whether already acquired or
hereafter acquired, that would be infringed by some manner, permitted
by this License, of making, using, or selling its contributor version,
but do not include claims that would be infringed only as a
consequence of further modification of the contributor version.  For
purposes of this definition, "control" includes the right to grant
patent sublicenses in a manner consistent with the requirements of
this License.

  Each contributor grants you a non-exclusive, worldwide, royalty-free
patent license under the contributor's essential patent claims, to
make, use, sell, offer for sale, import and otherwise run, modify and
propagate the contents of its contributor version.

  In the following three paragraphs, a "patent license" is any express
```

```
agreement or commitment, however denominated, not to enforce a patent
(such as an express permission to practice a patent or covenant not to
sue for patent infringement).  To "grant" such a patent license to a
party means to make such an agreement or commitment not to enforce a
patent against the party.

  If you convey a covered work, knowingly relying on a patent license,
and the Corresponding Source of the work is not available for anyone
to copy, free of charge and under the terms of this License, through a
publicly available network server or other readily accessible means,
then you must either (1) cause the Corresponding Source to be so
available, or (2) arrange to deprive yourself of the benefit of the
patent license for this particular work, or (3) arrange, in a manner
consistent with the requirements of this License, to extend the patent
license to downstream recipients.  "Knowingly relying" means you have
actual knowledge that, but for the patent license, your conveying the
covered work in a country, or your recipient's use of the covered work
in a country, would infringe one or more identifiable patents in that
country that you have reason to believe are valid.

  If, pursuant to or in connection with a single transaction or
arrangement, you convey, or propagate by procuring conveyance of, a
covered work, and grant a patent license to some of the parties
receiving the covered work authorizing them to use, propagate, modify
or convey a specific copy of the covered work, then the patent license
you grant is automatically extended to all recipients of the covered
work and works based on it.

  A patent license is "discriminatory" if it does not include within
the scope of its coverage, prohibits the exercise of, or is
conditioned on the non-exercise of one or more of the rights that are
specifically granted under this License.  You may not convey a covered
work if you are a party to an arrangement with a third party that is
in the business of distributing software, under which you make payment
to the third party based on the extent of your activity of conveying
the work, and under which the third party grants, to any of the
parties who would receive the covered work from you, a discriminatory
patent license (a) in connection with copies of the covered work
conveyed by you (or copies made from those copies), or (b) primarily
for and in connection with specific products or compilations that
contain the covered work, unless you entered into that arrangement,
or that patent license was granted, prior to 28 March 2007.

  Nothing in this License shall be construed as excluding or limiting
any implied license or other defenses to infringement that may
otherwise be available to you under applicable patent law.

  12. No Surrender of Others' Freedom.

  If conditions are imposed on you (whether by court order, agreement or
otherwise) that contradict the conditions of this License, they do not
excuse you from the conditions of this License.  If you cannot convey a
covered work so as to satisfy simultaneously your obligations under this
License and any other pertinent obligations, then as a consequence you may
not convey it at all.  For example, if you agree to terms that obligate you
to collect a royalty for further conveying from those to whom you convey
the Program, the only way you could satisfy both those terms and this
```

License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License ”or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM ”AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF

```
DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD
PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS),
EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF
SUCH DAMAGES.

  17. Interpretation of Sections 15 and 16.

  If the disclaimer of warranty and limitation of liability provided
above cannot be given local legal effect according to their terms,
reviewing courts shall apply local law that most closely approximates
an absolute waiver of all civil liability in connection with the
Program, unless a warranty or assumption of liability accompanies a
copy of the Program in return for a fee.

                     END OF TERMS AND CONDITIONS

            How to Apply These Terms to Your New Programs

  If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this is to make it
free software which everyone can redistribute and change under these terms.

  To do so, attach the following notices to the program.  It is safest
to attach them to the start of each source file to most effectively
state the exclusion of warranty; and each file should have at least
the ”copyright” line and a pointer to where the full notice is found.

    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

  If the program does terminal interaction, make it output a short
notice like this when it starts in an interactive mode:

    <program>  Copyright (C) <year>  <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type ‘show c’ for details.

The hypothetical commands ‘show w’ and ‘show c’ should show the appropriate
parts of the General Public License.  Of course, your program's commands
might be different; for a GUI interface, you would use an ”about box”.
```

```
  You should also get your employer (if you work as a programmer) or school,
if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see
<http://www.gnu.org/licenses/>.

  The GNU General Public License does not permit incorporating your program
into proprietary programs.  If your program is a subroutine library, you
may consider it more useful to permit linking proprietary applications with
the library.  If this is what you want to do, use the GNU Lesser General
Public License instead of this License.  But first, please read
<http://www.gnu.org/philosophy/why-not-lgpl.html>.
```

# 7 extra/.gitignore

```
ppmtoy4m_quickndirty
```

# 8 extra/Makefile

```
#!/usr/bin/make -f
#
# ppmtoy4m_quickndirty -- faster PPM to YUV4MPEG2 conversion
# Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#

GCC=gcc -std=c99 -Wall -pedantic -O3 -march=native

all: ppmtoy4m_quickndirty

clean:
	-rm -f ppmtoy4m_quickndirty

.SUFFIXES:
.PHONY: all clean

ppmtoy4m_quickndirty: ppmtoy4m_quickndirty.c
	$(GCC) -o ppmtoy4m_quickndirty ppmtoy4m_quickndirty.c
```

# 9 extra/ppmtoy4m_quickndirty.c

```
/*
ppmtoy4m_quickndirty -- faster PPM to YUV4MPEG2 conversion
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.


Portions of this code are derived from mjpegtools:
”mjpeg/mjpeg_play/lavtools/colorspace.c”

colorspace.c: Routines to perform colorspace conversions.
Copyright (C) 2001 Matthew J. Marjanovic <maddog@mir.com>

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.
*/

#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define FP_BITS 18

static inline int myround(double n) {
  if (n >= 0) return (int)(n + 0.5);
  else        return (int)(n - 0.5);
}

#define Y 0
#define U 1
#define V 2
#define R 0
#define G 1
#define B 2

int main(int argc, char **argv) {
  if (argc != 3) { return 1; }
  char *widths = argv[1];
  char *heights = argv[2];
  char ppmtemplate[1024];
  snprintf(ppmtemplate, 1000, "P6\n%s %s 255\n", widths, heights);
  int width;
  int height;
  if (sscanf(ppmtemplate, "P6\n%d %d 255\n", &width, &height) != 2) { fprintf(stderr, "ppm template parse\n"); return 1; }
  int n = width * height;
  unsigned char *rgbs = malloc(n * 3);
  unsigned char *ys = malloc(n);
  unsigned char *us = malloc(n);
  unsigned char *vs = malloc(n);
  char y4m[1024];
  snprintf(y4m, 1000, "YUV4MPEG2 W%d H%d F25:1 Ip A1:1 C444\n", width, height);
  if (fwrite(y4m, strlen(y4m), 1, stdout) != 1) { fprintf(stderr, "y4m header write\n"); return 1; }
  int32_t cc[3][3][256];
  for (int i = 0; i < 256; ++i) {
    cc[Y][R][i] = myround(0.299 * (double)i * 219.0 / 255.0 * (double)(1<<FP_BITS));
    cc[Y][G][i] = myround(0.587 * (double)i * 219.0 / 255.0 * (double)(1<<FP_BITS));
    cc[Y][B][i] = myround((0.114 * (double)i * 219.0 / 255.0 * (double)(1<<FP_BITS)) + (double)(1<<(FP_BITS-1)) + (16.0 * (double)(1<<FP_BITS)));
    cc[U][R][i] = myround(-0.168736 * (double)i * 224.0 / 255.0 * (double)(1<<FP_BITS));
    cc[U][G][i] = myround(-0.331264 * (double)i * 224.0 / 255.0 * (double)(1<<FP_BITS));
    cc[U][B][i] = myround((0.500 * (double)i * 224.0 / 255.0 * (double)(1<<FP_BITS)) + (double)(1<<(FP_BITS-1)) + (128.0 * (double)(1<<FP_BITS)));
    cc[V][R][i] = myround(0.500 * (double)i * 224.0 / 255.0 * (double)(1<<FP_BITS));
    cc[V][G][i] = myround(-0.418688 * (double)i * 224.0 / 255.0 * (double)(1<<FP_BITS));
    cc[V][B][i] = myround((-0.081312 * (double)i * 224.0 / 255.0 * (double)(1<<FP_BITS)) + (double)(1<<(FP_BITS-1)) + (128.0 * (double)(1<<FP_BITS)));
  }
  int ppml = strlen(ppmtemplate);
  char ppm[1024];
  while (fread(ppm, ppml, 1, stdin) == 1) {
    ppm[ppml] = '\0';
    if (strncmp(ppmtemplate, ppm, ppml)) { fprintf(stderr, "ppm template mismatch\n%s%s", ppmtemplate, ppm); return 1; }
    if (fread(rgbs, n * 3, 1, stdin) != 1) { fprintf(stderr, "ppm read mismatch\n"); return 1; }
    unsigned char *rgb = rgbs;
    unsigned char *y = ys;
    unsigned char *u = us;
    unsigned char *v = vs;
    for (int i = 0; i < n; ++i) {
      int r = *rgb++;
      int g = *rgb++;
      int b = *rgb++;
      *y++ = (cc[Y][R][r] + cc[Y][G][g] + cc[Y][B][b]) >> FP_BITS;
      *u++ = (cc[U][R][r] + cc[U][G][g] + cc[U][B][b]) >> FP_BITS;
```

```
        *v++ = (cc[V][R][r] + cc[V][G][g] + cc[V][B][b]) >> FP_BITS;
      }
      if (fwrite("FRAME\n", 6, 1, stdout) != 1) { fprintf(stderr, "y4m frame write↙
         ↳ \n"); return 1; }
      if (fwrite(ys, n, 1, stdout) != 1) { fprintf(stderr, "y4m y write\n"); ↙
         ↳ return 1; }
      if (fwrite(us, n, 1, stdout) != 1) { fprintf(stderr, "y4m u write\n"); ↙
         ↳ return 1; }
      if (fwrite(vs, n, 1, stdout) != 1) { fprintf(stderr, "y4m v write\n"); ↙
         ↳ return 1; }
    }
  return 0;
}
```

## 10 .gitignore

```
dist
ChangeLog
```

## 11 HACKING

```
Hacking Mandulia
================


Source Code Index
-----------------

src/                    - 'mandulia' program source code
config/                 - 'mandulia' runtime configurations

src/AmmannA3.hs         - zooming interface for a substitution tiling
src/Bounds.hs           - operations on bounding rectangles
src/Image.hs            - transport images from CPU to GPU
src/Interface.hs        - interface for scripting 'mandulia' in Lua
src/JobQueue.hs         - a dynamically reprioritizable todo list
src/Julia.hs            - worker thread for rendering Julia Sets
src/Mandulia.hs         - main program (this file needs to be split up)
src/PriorityCache.hs    - for caching the N best-right-now items
src/qsort.c             - wrapper for C's qsort() (code implementations)
src/qsort.h             - wrapper for C's qsort() (data definitions)
src/ResourcePool.hs     - allocate up to N resources, reusing them
src/rjulia.c            - render Julia Sets (code implementations)
src/rjulia.h            - render Julia Sets (code specifications)
src/rts.c               - configure GHC default runtime options
src/Snapshot.hs         - export the OpenGL frame buffer to PPM
src/Sort.hsc            - interface with 'qsort.c' and 'qsort.h'
src/StatsLogger.hs      - runtime statistics logging mechanism
src/TextureCache.hs     - cache the N best-right-now textures on the GPU
src/Utils.hs            - miscellaneous general purpose utility functions
src/Vector.hs           - linear algebra ((2+1)D homegenous coordinates)

config/defaults.lua     - default settings (to be used by configurations)
config/distance.lua     - compute Mandelbrot exterior distance (used by 'main')
config/main.lua         - 'main' configuration (interact and attract modes)
config/transition.lua   - interpolate between viewports (used by 'main')
```

## 12 INSTALL

```
Installation
============

The easy way
------------

Get 'gcc' and 'cabal-install' and run 'cabal install'.
If your CPU does not support SSE4 instructions, add '-f-SSE4'.


The slightly less easy way
--------------------------

If you cannot 'cabal-install', follow the usual Cabal procedure:

    runhaskell Setup.hs configure
    runhaskell Setup.hs build
    runhaskell Setup.hs install

Note that you will need 'GLUT' and 'hslua' installed first.

For more information:

    runhaskell Setup.hs help


See Also
--------

http://www.haskell.org/cabal/
```

## 13 loc.sh

```
#!/bin/bash
#
# Mandulia -- Mandelbrot/Julia explorer
# Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>
#
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.
#
blines=18
bwords=111
bchars=728
count=0
```

```
wc src/* config/* |
while read lines words chars filename
do
  if [[ ”${filename}” == ”total” ]]
  then
    lines=$(( lines - count * blines ))
    words=$(( words - count * bwords ))
    chars=$(( chars - count * bchars ))
    echo ”Source code statistics: ${lines} lines, ${words} words, ${chars} chars↙
       ↪ .”
  else
    count=$(( count + 1 ))
  fi
done
```

## 14 mandulia.cabal

```
Name:                  mandulia
Version:               0.7.0.1
Stability:             Provisional
Synopsis:              A zooming visualisation of the Mandelbrot Set as many Julia↙
    ↪ Sets.
Description:           Mandulia provides a zooming visualisation of the Mandelbrot↙
    ↪ Set
                       as many Julia Sets.  Featuring a profiled and optimized ↙
                           ↪ renderer,
                       and a Lua configuration and scripting interface.
                       .
                       Mandulia supports additional compile-time flags for speed, ↙
                           ↪ but by
                       default these are disabled because they don't work on all ↙
                           ↪ systems.
                       These flags are ”-ffast” for miscellaneous optimizations, ↙
                           ↪ and
                       ”-ffast -fSSE4” to use SSE4 instructions too.
                       .
                       Changes from ”mandulia-0.6”: the optimization flags are ↙
                           ↪ disabled
                       by default as they were causing problems on some systems; ↙
                           ↪ reduced
                       visual jitter for smoother animation; smoother colour ↙
                           ↪ gradations;
                       access runtime statistics from Lua scripts.
Homepage:              http://code.mathr.co.uk/mandulia
Cabal-version:         >=1.6
License:               GPL-3
License-file:          COPYING
Author:                Claude Heiland-Allen
Maintainer:            claude@mathr.co.uk
Copyright:             (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>
Category:              Graphics
Build-type:            Simple
Extra-source-files:    AUTHORS ChangeLog HACKING INSTALL NEWS README TODO
                       src/rjulia.h src/qsort.h
Data-dir:              config
Data-files:            defaults.lua distance.lua main.lua transition.lua
```

```
Flag fast
  Description:          Enable optimizations that might break some compilers.
  Default:              False

Flag SSE4
  Description:          Enable optimizations for SSE4 CPUs.
  Default:              False

Executable mandulia
  Build-depends:        base >= 4 && < 5, array, bytestring, containers, directory, filepath, time, GLUT >= 2.2, hslua >= 0.2
  Build-tools:          hsc2hs
  Extensions:           ForeignFunctionInterface
  if !flag(fast) && impl(ghc)
    GHC-options:          -Wall -threaded
    GHC-prof-options:     -Wall -threaded -prof -auto-all
    CC-options:           -std=c99 -Wall -pedantic
  if  flag(fast) &&  flag(SSE4) && impl(ghc)
    GHC-options:          -Wall -O3 -threaded -fvia-c -funbox-strict-fields -optc-O3 -optc-march=native -optc-ffast-math -optc-msse4
    GHC-prof-options:     -Wall -O3 -threaded -fvia-c -funbox-strict-fields -optc-O3 -optc-march=native -optc-ffast-math -optc-msse4 -prof -auto-all
    CC-options:           -std=c99 -Wall -pedantic -O3 -march=native -ffast-math -msse4
  if  flag(fast) && !flag(SSE4) && impl(ghc)
    GHC-options:          -Wall -O3 -threaded -fvia-c -funbox-strict-fields -optc-O3 -optc-march=native -optc-ffast-math
    GHC-prof-options:     -Wall -O3 -threaded -fvia-c -funbox-strict-fields -optc-O3 -optc-march=native -optc-ffast-math -prof -auto-all
    CC-options:           -std=c99 -Wall -pedantic -O3 -march=native -ffast-math
  HS-source-dirs:       src
  Main-is:              Mandulia.hs
  Other-modules:        AmmannA3
                        Bounds
                        Image
                        Interface
                        JobQueue
                        Julia
                        Mandulia
                        PriorityCache
                        ResourcePool
                        Snapshot
                        Sort
                        StatsLogger
                        TextureCache
                        Utils
                        Vector
  C-sources:            src/rjulia.c
                        src/rts.c
                        src/qsort.c
  Include-dirs:         src
  Includes:             rjulia.h
                        qsort.h

Source-repository head
  type:                 git
  location:             http://code.mathr.co.uk/mandulia.git
```

```
Source-repository this
  type:                 git
  location:             http://code.mathr.co.uk/mandulia.git
  tag:                  v0.7.0.1
```

## 15 NEWS

```
v0.7 2010-08-14 smooth-as

    Visual quality improvements include reduced jitter (at the cost of
    additional latency of one frame) and smoother colour gradations.
    Portability improvements include disabled optimizations by default,
    add -ffast or -ffast -fSSE4 to re-enable them.  Feature enhancements
    include access of runtime statistics from Lua scripts (for example:
    mandulia.statistics.frame.stddev).

    Source code statistics: 2097 lines, 10174 words, 59218 chars.

v0.6 2010-07-25 facepalm

    The previous version had a bug wherein the program help text was out
    of sync with the implementation, leading to mass confusion and such.
    This version fixes that bug.

    Source code statistics: 2071 lines, 10047 words, 58373 chars.

v0.5 2010-07-25 neon in O(1) space

    This version runs in asymptotic constant space.  Aesthetics tweaks
    give more colourful 'neon' output, and the randomized attract mode
    picks more interesting places to explore.

    Source code statistics: 2072 lines, 10050 words, 58399 chars.

v0.4 2010-07-12 joining the cabal

    This version is installable with 'cabal-install'.  Documentation has
    been improved, many bugs have been fixed, and new features have been
    added, including:

      * full screen mode
      * argument passing to configuration scripts
      * a search heirarchy for configuration scripts
      * a separate namespace in the scripting API

    Source code statistics: 1948 lines, 9626 words, 55247 chars.

v0.3 2010-07-12 scripting interface

    This version features the new feature of external configuration
    scripts in Lua.  The scripts can configure initial settings as well
    as provide for interaction (via keyboard) and animation, and more
    core features will be exposed to the scripting interface as versions
    progress.

    Source code statistics: 1954 lines, 9774 words, 56087 chars.
```

```

v0.2 2010-07-10 optimized and improved

    This version features greatly improved speed thanks to hours of
    careful profiling, along with aesthetic enhancements and runtime
    statistics logging.  Some bugs were fixed, and video recording is
    present (though disabled by default).

v0.1 2010-07-06 first working version

    Mandulia provides a zooming visualisation of the Mandelbrot Set as
    many Julia Sets.  This first version, while working mostly as
    intended, has too many hard-coded configurations and some code
    ugliness.

v0.0 2010-07-02 initial commit

    Project started.
```

## 16 README

```
Mandulia
========

Mandelbrot/Julia set explorer.


Invocation
----------

Multi-core CPU with lots of RAM recommended:

    mandulia configuration

Currently there is one preset configurations available:
'main' -- keyboard navigation with randomized attract mode.

To render videos (see below):

    mandulia configuration |
    ppmtoy4m -S444 -F25:1 |
    ffmpeg2theora -v 8 -f yuv4mpegpipe -o mandulia.ogv -


Controls
--------

'main' defines the following key bindings:

Escape   -- quit
F11      -- toggle full screen
a        -- enter attract mode immediately
r        -- record for 10 minutes (assuming 25fps)
s        -- record for 1 frame (aka screenshot)

Note that recording is via PPM images on 'stdout', so to avoid flooding
your terminal with binary data be sure to redirect it somewhere else.
```

```
As a side-effect, the following count as 'activity':

PageUp    -- zoom in
PageDown  -- zoom out
cursors   -- move around
End       -- stop moving
Home      -- return to base
[ ]       -- change speed
#         -- reset speed
{ }       -- change weight
~         -- reset weight

Attract mode is entered after 1 minute (assuming 25fps) of inactivity,
and left on any activity.

Other configurations may differ, check their sources for details.


Scripting
---------

Mandulia can be reconfigured without recompilation, using Lua scripts.
The first (non RTS) argument to 'mandulia' will be executed on startup.
First it will try to load it as a file, then via Lua's 'require()'.
See 'defaults.lua' to see the available interface, and 'main.lua' for an
example of keyboard navigation. Note that some variables only take
effect on startup, and later modification will be ignored.
```

## 17 sdist.sh

```
#!/bin/bash
(
  echo "ChangeLog exported from Git by ${USER}@${HOSTNAME} at $(date --iso=s)"
  git log --pretty=tformat:"%d~ %h %ai %s" |
  sed "s|^ (|\n (|" |
  sed "s|)~|)\n\n~|" |
  sed "s|^~ ||" |
  sed "s|; |\n                                      |g"
  echo ""
  echo " (eof)"
) > ChangeLog
cabal sdist
```

## 18 Setup.hs

```
import Distribution.Simple
main = defaultMain
```

## 19 src/AmmannA3.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
```

```
-}

module AmmannA3 (AmmannA3(), ammannA3, Tile(..), Tile'(..), tiles, zoom, zoomTo,↙
    ↳ stepIn) where

import Data.Maybe (fromMaybe)
import Data.Tree

import Bounds
import Utils
import Vector

data Tile = A | B | C
  deriving (Show, Read, Eq, Ord, Enum, Bounded)

bounds0 :: Tile -> Bounds
bounds0 x = bounds0' !! fromEnum x

bounds0' :: [Bounds]
bounds0' = [ sbound 1 phi' -- A
           , sbound 2 1    -- B
           , sbound 1 1    -- C
           ]

sbound :: R -> R -> Bounds
sbound w h =
  bounds [ V x y 1 | x <- [negate w, w], y <- [negate h, h] ]

transforms :: [( Tile, [( Tile, (M, Integer) )] )]
transforms =
  [ (A, [ (B, (t ( 0) (p * n/2 - f/2)                    0 ↙
      ↳                                  , 1))
        , (A, (t (-1) (f/2 - p * e/2)                    0 ↙
            ↳                                  , 2)) ])
  , (B, [ (A, (t ( 1) (p * e/2               - n/2)   (p * f/2 - k/2 - m/2) ↙
      ↳          , 3))
        , (C, (t ( 2) (p * (a + x/2 + v/2)   - n/2)   (p * (s/2 + u/2) - k/2 - m↙
            ↳ /2) , 4))
        , (A, (t (-1) (p * (a + v - c + e/2) - n/2)   (k/2 + m/2          - p * f↙
            ↳ /2) , 5))
        , (A, (t ( 0) (n/2                   - p * f/2)   (p * e/2          - k/2 - m↙
            ↳ /2) , 6)) ])
  , (C, [ (C, (t ( 2) (p * (x/2 + v/2) - x/2 - v/2)   (p * (s/2 + u/2) - s/2 - u↙
      ↳ /2) , 7))
        , (A, (t (-1) (p * (v - c + e/2) - x/2 - v/2) (s/2 + u/2 - p * f/2) ↙
            ↳          , 8))
        , (A, (t ( 0) (x/2 + v/2 - p * f/2)          (p * e/2        - s/2 - u↙
          ↪ /2)  , 9)) ])
  ]
  where
    t da dx dy = translate (dx*2) (dy*2) ^^*^^ rotate (da * pi / 2) ^^*^^ scale ↙
      ↪ p p
    p = phi'
    a = p * p
    c = p * p * p
    e = p
    f = 1
    k = p
    m = p * p
    n = (1 - p * p * p) / p
    s = p
    u = p * p
    v = p * p * p + p
    x = p * p * p * p

centerC :: V
centerC =
  let Just ts     = lookup C transforms
      Just (t,_) = lookup C ts
      ps = iterate (t ^^*^) (V 0 0 1)
  in  ps !! 256

inRadiusC :: R
inRadiusC =
  let cornerC = V (1/2 - phi') (1/2 - phi' * phi') 1
  in  cornerC ^|-|^ centerC

data Tile'' =
  Tile''
    { ttTile      :: !Tile
    , ttId        :: !Integer
    , ttTransform :: !M
    }

builder :: Tile'' -> (Tile'', [Tile''])
builder tm = tm ‘seq‘ (tm, map mkTile (mine transforms))
  where
    mine = concatMap snd . filter ((==) (ttTile tm) . fst)
    mkTile (x, (mm, j)) =
      Tile''
        { ttTile = x
        , ttId = 10 * ttId tm + j
        , ttTransform = ttTransform tm ^^*^^ mm
        }

data Tile' =
  Tile'
    { tTile   :: !Tile
    , tBounds :: !Bounds
    , tCenter :: !V
    , tDepth  :: !Int
    , tLevel  :: !Int
    , tId     :: !Integer
    }

tree :: R -> Tree Tile'
tree maxRadius =
  let s = maxRadius / inRadiusC
      V x y _ = centerC
      tr = scale s s ^^*^^ translate (-x) (-y)
      t0 = Tile''{ ttTile = C, ttId = 7, ttTransform = tr }
  in toTiles (Just (V 0 0 1)) C 0 (tree' t0)

tree' :: Tile'' -> Tree Tile''
tree' t = unfoldTree builder t

tB :: M -> Tile -> Bounds
tB m t = m `transform'` bounds0 t

toTiles :: Maybe V -> Tile -> Int -> Tree Tile'' -> Tree Tile'
toTiles v0 t0 level tr =
  let Tile''{ ttTile = t, ttId = n, ttTransform = m } = rootLabel tr
      ts  = subForest tr
      v1 = fromMaybe (centerPoint m) v0
      v2 = if t0 == C && t == C then v0 else Nothing
      b1 = tB m t
      nn = normalizeId n
      tile =
        Tile'
          { tTile = t
          , tBounds = b1
          , tCenter = v1
          , tDepth  = idToLevel' nn
          , tLevel  = level
          , tId = nn
          }
      level' = level + 1
      forest = level' `seq` map (toTiles v2 t level') ts
  in tile `seq` Node{ rootLabel = tile, subForest = forest }

centerPoint :: M -> V
centerPoint = (^^*^ centerC)

data LevelA3 =
  LevelA3
    { lInnerTiles :: Forest Tile'
    , lOuterTiles :: Forest Tile'
    , lBounds     :: Bounds
    }

data AmmannA3 =
  AmmannA3
    { aOuter  :: [LevelA3]
    , aFocus  ::  LevelA3
    , aBounds :: Bounds
    , aRadius :: R
    }

ammannA3 :: Bounds -> AmmannA3
ammannA3 box =
  let r = diagonal box / 2
      (is, os, _) = triPart box [tree r]
      l = LevelA3{ lInnerTiles = is, lOuterTiles = os, lBounds = box }
  in  AmmannA3{ aOuter = [l], aFocus = l, aBounds = box, aRadius = r }

triPart :: Bounds -> [Tree Tile'] -> ([Tree Tile'], [Tree Tile'], [Tree Tile'])
triPart box = foldr go ([],[],[])
  where
    go t (is, es, os)
      | b `insideOrEqual` box = (t:is, es, os)
      | b `outside`       box = (is, es, t:os)
      | otherwise             = (is, t:es, os)
      where b = tBounds . rootLabel $ t

zoomTo :: Bounds -> AmmannA3 -> Maybe AmmannA3
zoomTo box a3
  | box `insideOrEqual` region =
    (if factor >= phi'
      then Just
      else zoomTo box . (\a -> a{ aRadius = phi' * aRadius a }) . stepIn) $
        let focus = aFocus a3
            ots   = prune box (lOuterTiles focus)
            (its, ots', _) = triPart box (lInnerTiles focus)
        in  a3{ aFocus = focus{ lOuterTiles = ots' ++ ots
                              , lInnerTiles = its
                              , lBounds = box
                              } }
  | otherwise = zoomTo box =<< stepOut a3{ aRadius = phi  * aRadius a3 }
  where
    factor = radius / aRadius a3
    radius = diagonal box / 2
    region = lBounds . aFocus $ a3

zoom :: R -> AmmannA3 -> Maybe AmmannA3
zoom factor a3 = flip zoomTo a3 . expand factor . lBounds . aFocus $ a3

prune :: Bounds -> Forest Tile' -> Forest Tile'
prune box = filter (not . outside box . tBounds . rootLabel)

stepOut :: AmmannA3 -> Maybe AmmannA3
stepOut a3 =
  case aOuter a3 of
    []     -> Nothing
    os@[l] -> Just a3{ aOuter = os, aFocus = l, aRadius = aRadius a3 * phi' }
    (l:os) -> Just a3{ aOuter = os, aFocus = l }

stepIn  :: AmmannA3 -> AmmannA3
stepIn  a3 =
  let l0  =     aFocus a3
      os  = l0:aOuter a3
      box = lBounds l0
      its =                          concatMap subForest . lInnerTiles $ l0
      (its', ots, _) = triPart box . concatMap subForest . lOuterTiles $ l0
      l   = l0{ lInnerTiles = its' ++ its, lOuterTiles = ots }
  in a3{ aOuter = os, aFocus = l }

tiles :: Int -> AmmannA3 -> [Tile']
tiles lod = map rootLabel . (\l -> lOuterTiles l ++ lInnerTiles l) . aFocus . ↙
    ↳ (!!lod) . iterate stepIn

normalizeId :: Integer -> Integer -- C=>C is transform 7
normalizeId n = let (d, m) = n `divMod` 10 in if m == 7 then normalizeId d else ↙
    ↳ n

idToLevel' :: Integer -> Int -- n must be normalized
idToLevel' n = snd . head . dropWhile ((<n) . fst) $ tens

tens :: [(Integer, Int)]
tens = iterate (10*) 1 `zip` [0..]
```

## 20 src/Bounds.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module Bounds(
  Bounds(), bounds, corners, center,
  bottomLeft, bottomRight, topLeft, topRight,
  left, right, above, below,
  leftOrEqual, rightOrEqual, aboveOrEqual, belowOrEqual,
  outside, inside, insideOrEqual, overlap,
  transform, transform', expand, diagonal,
  into
) where

import Data.List (foldl')

import Vector

data Bounds = Bounds{ bottomLeft :: !V, topRight :: !V }
  deriving (Show, Read, Eq, Ord)

topLeft :: Bounds -> V
topLeft box =
  let V x _ _ = bottomLeft box
      V _ y _ = topRight   box
  in  V x y 1

bottomRight :: Bounds -> V
```

```haskell
bottomRight box =
  let V x _ _ = topRight   box
      V _ y _ = bottomLeft box
  in  V x y 1

bounds :: [V] -> Bounds
bounds [] = error "Bounds.bounds []"
bounds (V u v _ : vs) =
  let f (a, b, c, d) (V x y _) = (min a x, max b x, min c y, max d y)
      (x0, x1, y0, y1) = foldl' f (u, u, v, v) vs
  in  Bounds{ bottomLeft = V x0 y0 1, topRight = V x1 y1 1 }

corners :: Bounds -> [V]
corners box =
  map ($ box) [topLeft, topRight, bottomLeft, bottomRight]

center :: Bounds -> V
center box = (bottomLeft box ^+^ topRight box) ^/ 2

expand :: R -> Bounds -> Bounds
expand z box =
  let c = center box
      t v = ((v ^-^ c) ^* z) ^+^ c
  in  bounds . map (t . ($ box)) $ [bottomLeft, topRight]

left  :: V -> V -> Bool
left  (V u _ _) (V x _ _) = u < x

right :: V -> V -> Bool
right (V u _ _) (V x _ _) = u > x

above :: V -> V -> Bool
above (V _ v _) (V _ y _) = v > y

below :: V -> V -> Bool
below (V _ v _) (V _ y _) = v < y

leftOrEqual  :: V -> V -> Bool
leftOrEqual  (V u _ _) (V x _ _) = u <= x

rightOrEqual :: V -> V -> Bool
rightOrEqual (V u _ _) (V x _ _) = u >= x

aboveOrEqual :: V -> V -> Bool
aboveOrEqual (V _ v _) (V _ y _) = v >= y

belowOrEqual :: V -> V -> Bool
belowOrEqual (V _ v _) (V _ y _) = v <= y

outside :: Bounds -> Bounds -> Bool
outside box region =
  bottomLeft box `above` topRight   region ||
  bottomLeft box `right` topRight   region ||
  topRight   box `below` bottomLeft region ||
  topRight   box `left`  bottomLeft region

inside  :: Bounds -> Bounds -> Bool
inside  box region =
  bottomLeft box `above` bottomLeft region &&
  bottomLeft box `right` bottomLeft region &&
  topRight   box `below` topRight   region &&
  topRight   box `left`  topRight   region

insideOrEqual :: Bounds -> Bounds -> Bool
insideOrEqual box region =
  bottomLeft box `aboveOrEqual` bottomLeft region &&
  bottomLeft box `rightOrEqual` bottomLeft region &&
  topRight   box `belowOrEqual` topRight   region &&
  topRight   box `leftOrEqual`  topRight   region

overlap :: Bounds -> Bounds -> Bool
overlap box region =
  not (box `inside` region || box `outside` region)

transform :: M -> Bounds -> Bounds
transform m = bounds . map (m ^^*^) . corners

-- transform' precondition: m's rotation is a multiple of pi/2
transform' :: M -> Bounds -> Bounds
transform' m bs = bounds [ m ^^*^ bottomLeft bs, m ^^*^ topRight bs ]

diagonal :: Bounds -> R
diagonal box = topRight box ^|-|^ bottomLeft box

into :: Bounds -> Bounds -> M
into box region =
  let V x0 y0 _ = center box
      V x1 y1 _ = center region
      s = diagonal region / diagonal box
  in  translate x1 y1 ^^*^^ scale s s ^^*^^ translate (-x0) (-y0)
```

## 21 src/Image.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module Image(Image(), iWidth, iHeight, iChannels, iBuffer, image, upload) where

import Foreign.Ptr(Ptr())
import Foreign.Marshal.Alloc(mallocBytes)

import Graphics.UI.GLUT

data Image =
  Image
    { iWidth     :: Int
    , iHeight    :: Int
    , iChannels  :: Int
    , iBuffer    :: Ptr ()
    }

image :: Int -> Int -> Int -> IO Image
image w h c
  | w > 0 && h > 0 && c > 0 = do
    b <- mallocBytes $ w * h * c
    return Image{ iWidth = w, iHeight = h, iChannels = c, iBuffer = b }
  | otherwise = error $ "Image.image: " ++ show [w,h,c]

upload :: Image -> IO TextureObject
upload i
  | iChannels i == 4 = do
    [tex] <- genObjectNames 1
    texture Texture2D $= Enabled
    textureBinding Texture2D $= Just tex
    build2DMipmaps Texture2D RGBA' (fromIntegral $ iWidth i) (fromIntegral $ ↙
       ↳ iHeight i) (PixelData RGBA UnsignedByte (iBuffer i))
    textureFilter  Texture2D $= ((Linear', Just Linear'), Linear')
    textureWrapMode Texture2D S $= (Repeated, ClampToEdge)
    textureWrapMode Texture2D T $= (Repeated, ClampToEdge)
    textureBinding Texture2D $= Nothing
    texture Texture2D $= Disabled
    return tex
  | otherwise = error $ "Image.upload: " ++ show (iChannels i)
```

## 22 src/Interface.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module Interface (Interface(..), interface, closeInterface, reshape, render, ↙
    ↳ keyboard, atexit) where
```

```

import Control.Exception (try, IOException)
import Control.Monad (forM_, liftM3)
import Data.IORef
import System.FilePath ((</>), replaceFileName)
import System.Directory (getAppUserDataDirectory)

import Scripting.Lua (LuaState)
import qualified Scripting.Lua as Lua

import Paths_mandulia (getDataFileName)

-- interface

data Interface =
  Interface
    { iLua          :: !LuaState -- initialized in interface
    , iQuit         :: !Bool     -- can only be set by Lua 'quit'
    -- these are all updated from Lua globals in Haskell 'update'
    , iRecord       :: !Bool
    , iWidth        :: !Int
    , iHeight       :: !Int
    , iFPS          :: !Double
    , iFullScreen   :: !Bool
    , iDetail       :: !Double
    , iDisplaySize  :: !Double
    , iJuliaSize    :: !Int
    , iJobs         :: !Int
    , iImages       :: !Int
    , iTextures     :: !Int
    , iWorkers      :: !Int
    , iView         :: !(Double, Double, Double)
    -- these are updated from the Haskell side, set in Lua in 'update'
    , iStatistics   :: ![(String, (Double, Double, Double))]
    }

-- constructor

interface :: FilePath -> [String] -> IO (Maybe (IORef Interface))
interface f args = do
  l <- Lua.newstate
  iR <- newIORef Interface
    { iLua = l
    , iQuit = False
    , iRecord = False
    , iWidth = 1920
    , iHeight = 1080
    , iFullScreen = False
    , iFPS = 25
    , iDetail = 11
    , iDisplaySize = 96
    , iJuliaSize = 256
    , iJobs = 1024
    , iImages = 512
    , iTextures = 2048
    , iWorkers = 2
    , iView = (0, 0, 0)
    , iStatistics = []
    }
  Lua.openlibs l
  path <- getDataFileName "?.lua"
  appp <- getAppUserDataDirectory "mandulia"
  paths <- do
    Lua.getglobal2 l "package.path"
    r <- Lua.peek l (-1)
    Lua.pop l 1
    return r
  let paths' = ("."  </> "?.lua") ++ ";" ++     -- current dir
               (appp </> "?.lua") ++ ";" ++     -- user dir
               path ++                          -- cabal dir
               maybe "" (";" ++) paths          -- preset
      setPath p = do
        Lua.getglobal l "package"
        Lua.pushstring l "path"
        Lua.pushstring l p
        Lua.settable l (-3)
        Lua.pop l 1
  setPath paths'
  Lua.newtable l
  Lua.pushstring l "quit"
  Lua.pushhsfunction l (quit iR)
  Lua.settable l (-3)
  Lua.pushstring l "args"
  Lua.newtable l
  forM_ ([(1::Int) .. ] `zip` args) $ \(i,a) -> do
    Lua.push l i
    Lua.push l a
    Lua.settable l (-3)
  Lua.settable l (-3)
  Lua.setglobal l "mandulia"
  r1 <- (try $ do
    r <- Lua.loadfile l f
    if r /= 0 then return r else do
      let rel = replaceFileName f "?.lua"       -- relative dir
      setPath $ rel ++ ";" ++ paths'
      Lua.pcall l 0 0 0) :: IO (Either IOException Int)
  r2 <- if r1 == Right 0 then return r1 else do
    _ <- Lua.callproc l "require" f
    return $ Right 0
  if r2 == Right 0 then update iR >> return (Just iR) else Lua.close l >> return Nothing

-- destructor

closeInterface :: IORef Interface -> IO ()
closeInterface iR = do
  l <- iLua `fmap` readIORef iR
  Lua.close l

-- update from Lua

update :: IORef Interface -> IO ()
update iR = do
  l <- iLua `fmap` readIORef iR
```

```haskell
  -- copy statistics from Haskell to Lua
  Lua.getglobal l "mandulia"
  Lua.push l "statistics"
  Lua.newtable l
  iStatistics `fmap` readIORef iR >>= (mapM_ $ \(s,(a,b,c)) -> do
    Lua.push l s
    Lua.newtable l
    forM_ (words "count mean stddev" `zip` [a,b,c]) $ \(n,x) -> do
      Lua.push l n
      Lua.push l x
      Lua.settable l (-3)
    Lua.settable l (-3) )
  Lua.settable l (-3)
  Lua.pop l 1 -- pop mandulia global
  -- copy settings from Lua to Haskell
  let g s = do
        Lua.getglobal2 l ("mandulia." ++ s)
        r <- Lua.peek l (-1)
        Lua.pop l 1
        return r
  record      <- g "record"
  width       <- g "width"
  height      <- g "height"
  fps         <- g "fps"
  full        <- g "fullscreen"
  detail      <- g "detail"
  displaysize <- g "displaysize"
  juliasize   <- g "juliasize"
  jobs        <- g "jobs"
  images      <- g "images"
  textures    <- g "textures"
  workers     <- g "workers"
  viewX       <- g "view.x"
  viewY       <- g "view.y"
  viewZ       <- g "view.z"
  modifyIORef' iR $ \i ->
    let f k v = case v of
                  Nothing -> k i
                  Just x  -> x
    in   i { iRecord      = f iRecord      record
           , iWidth       = f iWidth       width
           , iHeight      = f iHeight      height
           , iFPS         = f iFPS         fps
           , iFullScreen  = f iFullScreen  full
           , iDetail      = f iDetail      detail
           , iDisplaySize = f iDisplaySize displaysize
           , iJuliaSize   = f iJuliaSize   juliasize
           , iJobs        = f iJobs        jobs
           , iImages      = f iImages      images
           , iTextures    = f iTextures    textures
           , iWorkers     = f iWorkers     workers
           , iView        = case liftM3 (,,) viewX viewY viewZ of
                              Just v -> v
                              _      -> iView i
           }

-- callbacks from Lua
```

```

quit :: IORef Interface -> IO ()
quit iR = modifyIORef' iR $ \i -> i{ iQuit = True }

-- callbacks into Lua

reshape :: IORef Interface -> Int -> Int -> IO ()
reshape iR w h = do
  l <- iLua ‘fmap‘ readIORef iR
  _ <- Lua.callproc l ”mandulia.reshape” w h
  update iR

render :: IORef Interface -> IO ()
render iR = do
  l <- iLua ‘fmap‘ readIORef iR
  _ <- Lua.callproc l ”mandulia.render”
  update iR

keyboard :: IORef Interface -> String -> IO ()
keyboard iR s = do
  l <- iLua ‘fmap‘ readIORef iR
  _ <- Lua.callproc l ”mandulia.keyboard” s
  update iR

atexit :: IORef Interface -> IO ()
atexit iR = do
  l <- iLua ‘fmap‘ readIORef iR
  _ <- Lua.callproc l ”mandulia.atexit”
  update iR
```

## 23 src/JobQueue.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module JobQueue(JobQueue(), jobQueue, getJob, putJob, withJob, completed, reprioritise) where

import Control.Concurrent.MVar
import Data.List (delete)
import Data.Maybe (maybeToList)
```

```haskell
type JobQueue j = MVar (JobQueue' j)

data JobQueue' j =
  JobQueue'
    { qDone    :: [j]
    , qPending :: [j]
    , qTodo    :: [j]
    , qNext    :: MVar j
    }

jobQueue :: IO (JobQueue j)
jobQueue = do
  nj <- newEmptyMVar
  newMVar JobQueue'{ qDone = [], qPending = [], qTodo = [], qNext = nj }

getJob :: JobQueue j -> IO j
getJob s = do
  q <- takeMVar s
  next <- tryTakeMVar $ qNext q
  case next of
    Nothing ->
      case qTodo q of
        []     -> do
          putMVar s q
          j <- takeMVar $ qNext q
          modifyMVar_ s $ \q' -> return q'{ qPending = j : qPending q' }
          return j
        (j:js) -> do
          putMVar s q{ qTodo = js }
          putMVar (qNext q) j
          j' <- takeMVar $ qNext q
          modifyMVar_ s $ \q' -> return q'{ qPending = j' : qPending q' }
          return j'
    Just j ->     do
          putMVar s q{ qPending = j : qPending q }
          return j

putJob :: Eq j => JobQueue j -> j -> IO ()
putJob s j = do
  q <- takeMVar s
  putMVar s q{ qDone = j : qDone q, qPending = delete j (qPending q) }

withJob :: Eq j => JobQueue j -> (j -> IO j) -> IO ()
withJob s action = putJob s =<< action =<< getJob s

completed :: JobQueue j -> IO [j]
completed s = do
  q <- takeMVar s
  let js = qDone q
  putMVar s q{ qDone = [] }
  return js

reprioritise :: Eq j => JobQueue j -> ([j] -> [j]) -> IO ()
reprioritise s f = do
  q <- takeMVar s
  j0 <- tryTakeMVar $ qNext q
  let jobs = filter (`notElem` qDone q ++ qPending q) (f (maybeToList j0 ++ ↙
      ↳ qTodo q))
  case jobs of
    []     -> do
      putMVar s q{ qTodo = [] }
    (j:js) -> do
      putMVar (qNext q) j
      putMVar s q{ qTodo = js }
```

## 24 src/Julia.hs

```
{-# LANGUAGE ForeignFunctionInterface #-}
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module Julia(Julia(..), score, JuliaJob(..), juliaWorker) where

import Control.Monad (forever)
import Foreign.Ptr (Ptr)
import Foreign.C.Types

import Graphics.UI.GLUT

import Image
import JobQueue
import ResourcePool
import StatsLogger(time, Logger)
import Utils

data Julia =
  Julia
    { jId    :: Integer
    , jLevel :: Int
    , jCX    :: Double
    , jCY    :: Double
    }
  deriving (Show, Read)

instance Eq Julia where
  j == k = jId j == jId k

instance Ord Julia where
  j `compare` k = jId j `compare` jId k
```

```
score :: Double -> Double -> Double -> Julia -> Double
score level cx cy j =
  let dcx = cx - jCX j
      dcy = cy - jCY j
      r2 = dcx * dcx + dcy * dcy
      dl = fromIntegral (jLevel j) - level
  in  sqrt r2 * phi ** dl
  -- see also: Poincaré half-plane distance metric
  -- 2 * atanh ( magnitude (z1 - z2) / magnitude (z1 - conjugate z2) )
  -- http://en.wikipedia.org/wiki/Poincar%C3%A9_metric
  -- #Metric_and_volume_element_on_the_Poincar.C3.A9_plane

data JuliaJob =
  JuliaJob
    { jCoords :: Julia
    , jDoneAction :: Maybe (Either (Julia, TextureObject) (Julia, (IO TextureObject, IO ())))
    }

instance Eq JuliaJob where
  j == k = jCoords j == jCoords k

foreign import ccall safe "rjulia.h julia_new"
  c_juliaNew     :: CInt -> CInt -> IO (Ptr ())

{-
foreign import ccall safe "rjulia.h julia_delete"
  c_juliaDelete :: Ptr () -> IO ()
-}

foreign import ccall safe "rjulia.h julia"
  c_julia        :: Ptr () -> Ptr () -> CDouble -> CDouble -> IO ()

juliaWorker :: Logger -> Int -> Int -> ResourcePool Image -> JobQueue JuliaJob -> IO ()
juliaWorker logStats w h is js = do
  c <- c_juliaNew (fromIntegral w) (fromIntegral h)
  forever $ do
    i <- acquire is
    withJob js $ \j -> do
      (dtC,()) <- time $ c_julia c (iBuffer i) (realToFrac . jCX . jCoords $ j) (realToFrac . jCY . jCoords $ j)
      logStats "compute" dtC
      return j
        { jDoneAction = Just . Right $ (jCoords j,
            ( do
                t <- upload i
                release is i
                return t
            , release is i
            ) )
        }
```

## 25 src/Mandulia.hs

```
{-
```

```
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module Main (main) where

import Control.Concurrent (ThreadId(), forkIO, killThread)
import Control.Monad (replicateM, when)
import Data.Either (partitionEithers)
import Data.IORef
import Data.Maybe (isNothing, catMaybes)
import Data.Time (UTCTime, getCurrentTime, diffUTCTime)
import Data.Version (showVersion)
import System.Environment (getArgs)
import System.Exit (exitFailure, exitSuccess)
import System.IO (hPutStr, stderr, stdout)

import Graphics.UI.GLUT hiding (scale, translate, fullScreen)
import qualified Graphics.UI.GLUT as G
import Unsafe.Coerce (unsafeCoerce)

import Paths_mandulia (version)

import AmmannA3
import Bounds
import Image
import Interface (Interface(..), interface, closeInterface)
import qualified Interface as I
import JobQueue
import Julia
import ResourcePool
import Snapshot
import Sort
import StatsLogger
import TextureCache
import Utils
import Vector

data Mandulia =
  Mandulia
    { tiling        :: Maybe AmmannA3
    , viewMax       :: Bounds
    , view          :: Bounds
    , width         :: Int
    , height      :: Int
    , fullScreen  :: Bool
    , oldWidth    :: Int
    , oldHeight   :: Int
    , iface       :: IORef Interface
    , workers     :: [ThreadId]
    , jobs        :: JobQueue JuliaJob
    , textures    :: TextureCache
    , images      :: ResourcePool Image
    , logStats    :: String -> Double -> IO ()
    , getStats    :: IO [(String, (Double, Double, Double))]
    , frameTime   :: Maybe UTCTime
    }

main :: IO ()
main = do
  args <- getArgs
  let (opts, args') = span (\o -> "-" == take 1 o) args
  when ("--version" `elem` opts || "-V" `elem` opts) $ do
    putStr $ unlines
      [ "mandulia " ++ showVersion version
      , "Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>"
      , "This is free software; see the source for copying conditions.  There is NO"
      , "warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE."
      ]
    exitSuccess
  when ("--help" `elem` opts || "-h" `elem` opts || "-?" `elem` opts) $ do
    putStr $ unlines
      [ "Usage: mandulia [OPTION]... CONFIGURATION [ARGUMENT]..."
      , ""
      , "CONFIGURATION is considered (in this order):"
      , "  as a file to load directly;"
      , "  as a module to load from the current directory;"
      , "  as a module to load from the user settings directory;"
      , "  as a module to load from the global settings directory."
      , ""
      , "Configurations available in this package may include:"
      , "  main"
      , ""
      , "The ARGUMENT list is supplied to the selected configuration."
      , ""
      , "Options:"
      , "  +RTS [OPTION].. -RTS   options for the run time system"
      , "                         (Try 'mandulia +RTS -? -RTS' for help)"
      , "  -?, -h, --help         print this help text"
      , "  -V, --version          print program version"
      , ""
      , "Report bugs to <claude@mathr.co.uk>."
      ]
    exitSuccess
  when (null args') $ do
    hPutStr stderr $ unlines
      [ "No configuration specified."
      , "Try 'mandulia --help' for more information."
      ]
    exitFailure
  hPutStr stderr $ unlines
      [ "mandulia (GPLv3+) 2010 Claude Heiland-Allen <claude@mathr.co.uk>"
      ]
  mif <- interface (head args') (tail args')
  when (isNothing mif) $ do
    hPutStr stderr $ unlines
      [ "Configuration error."
      ]
    exitFailure
  let Just iR = mif
  i <- readIORef iR
  let jsize  = clamp 1 1024 $ iJuliaSize i -- FIXME check power of two
      imagen = 1 `max` iImages    i
      texn   = 1 `max` iTextures i
      workn  = 1 `max` iWorkers   i
      winW   = 1 `max` I.iWidth   i
      winH   = 1 `max` I.iHeight  i
      full   = I.iFullScreen i
      mspf   = 1 `max` (ceiling $ 1000 / (iFPS i `max` 0.01))
      view0  = mkView winW winH 0 0 0
  (logStats', getStats') <- statsLogger
  jobq       <- jobQueue
  imgpool    <- resourcePool (image jsize jsize 4) imagen
  let texcache = textureCache texn
  wtids      <- replicateM workn
    (forkIO $ juliaWorker logStats' jsize jsize imgpool jobq)
  manduliaR <- newIORef Mandulia { tiling = Nothing
                                 , viewMax = view0
                                 , view = view0
                                 , width = winW
                                 , height = winH
                                 , fullScreen = full
                                 , oldWidth = winW
                                 , oldHeight = winH
                                 , iface = iR
                                 , workers = wtids
                                 , jobs = jobq
                                 , textures = texcache
                                 , images = imgpool
                                 , logStats = logStats'
                                 , getStats = getStats'
                                 , frameTime = Nothing
                                 }
  initialWindowSize $= Size (fromIntegral winW) (fromIntegral winH)
  initialDisplayMode $= [RGBAMode, DoubleBuffered]
  _ <- getArgsAndInitialize
  _ <- createWindow "Mandulia"
  displayCallback       $=      display manduliaR
  reshapeCallback       $= Just (reshape manduliaR)
  keyboardMouseCallback $= Just (kmouse  manduliaR)
  addTimerCallback mspf $       timer   manduliaR
  mainLoop

mkView :: Int -> Int -> Double -> Double -> Double -> Bounds
mkView winW winH x y z =
  let w = fromIntegral winW
```

```haskell
      h = fromIntegral winH
      ax = if winW > winH then 1 else w / h
      ay = if winW < winH then 1 else h / w
      r = 16 * phi' ** z
      x0 = x - r * ax
      y0 = y - r * ay
      x1 = x + r * ax
      y1 = y + r * ay
  in  bounds [ V x0 y0 1, V x1 y1 1 ]

quit :: IORef Mandulia -> IO ()
quit mR = do
  m <- readIORef mR
  mapM_ killThread (workers m)
  I.atexit (iface m)
  closeInterface (iface m)
  exitSuccess

update :: IORef Mandulia -> IO (Julia -> Double)
update mR = do
  m0 <- readIORef mR
  s <- getStats m0
  modifyIORef' (iface m0) (\i -> i{ iStatistics = s })
  I.render (iface m0)
  i <- readIORef (iface m0)
  when (iQuit i) (quit mR)
  fullscreen mR (iFullScreen i)
  m <- readIORef mR
  let (x, y, z) = iView i
      v = mkView (width m) (height m) x y z
  if v `insideOrEqual` viewMax m
   then do
    case zoomTo v (ammannA3 $ viewMax m) of
      t@(Just _) -> do
        writeIORef mR m{ tiling = t, view = v }
        return $ score z x y
      Nothing -> return $ score z x y -- FIXME should never happen?
   else return $ score z x y -- FIXME what to do when out of range?

data Quad =
  Quad
    { quadX :: !R
    , quadY :: !R
    , quadR :: !R
    , quadT :: !TextureObject
    }

radius :: R -> R -> Int -> R
radius d z i = let x = d + z - fromIntegral i
                in  clamp 0 1 $ x * 4 / d -- FIXME configure the 4

quads :: IORef Mandulia -> IO ([Julia], [Quad])
quads mR = do
  m <- readIORef mR
  i <- readIORef (iface m)
  let w = fromIntegral (width m)
      h = fromIntegral (height m)
      s = iDisplaySize i
      window = bounds [ V (-s) (-s) 1, V (w+s) (h+s) 1]
      viewT  = view m `into` window
      ctiles = case tiling m of
        Just t -> filter ((==) C . tTile) . tiles (ceiling d) $ t
        Nothing -> []
      d = iDetail i
      (_, _, z) = iView i
      rads = map ((s *) . radius d z) [0 ..]
  return $ partitionEithers
        [ case mt of
            Nothing  -> Left j
            Just tex -> Right Quad{ quadX = x
                                  , quadY = y
                                  , quadR = r
                                  , quadT = tex
                                  }
        | t <- ctiles
        , let V cx cy _ = tCenter t
        , let V  x  y _ = viewT ^^*^ V cx cy 1
        , let ii = tId t
        , let l = tDepth t
        , let r = rads !! l
        , let j = Julia{ jId = ii, jLevel = l, jCX = cx, jCY = cy }
        , let mt = lookupTexture (textures m) j
        ]

r2gl :: R -> GLdouble
r2gl = unsafeCoerce -- FIXME there must be a better way...

drawQuad :: Quad -> IO ()
drawQuad Quad{ quadX = x0, quadY = y0, quadR = s, quadT = tex } = do
  let t x y = texCoord $ TexCoord2 (x :: GLdouble) (y :: GLdouble)
      v x y = vertex   $ Vertex2 (r2gl x) (r2gl y)
  textureBinding Texture2D $= Just tex
  renderPrimitive Quads $ do
    color $ Color3 1 1 (1::GLdouble)
    t 0 1 >> v (x0 - s) (y0 + s)
    t 0 0 >> v (x0 - s) (y0 - s)
    t 1 0 >> v (x0 + s) (y0 - s)
    t 1 1 >> v (x0 + s) (y0 + s)

display0 :: IORef Mandulia -> IO ()
display0 mR = do
  swapBuffers
  t1 <- getCurrentTime
  mt0 <- frameTime `fmap` readIORef mR
  modifyIORef' mR $ \m -> m{ frameTime = Just t1 }
  case mt0 of
    Nothing -> return ()
    Just t0 -> do
      let dt = realToFrac (diffUTCTime t1 t0)
      readIORef mR >>= \m -> logStats m "frame" dt
  curScore <- update mR
  qs <- computeJobs mR curScore
  drawQuads qs
  completeJobs mR curScore
  record mR
  reportErrors

display :: IORef Mandulia -> IO ()
display mR = do
  m <- readIORef mR
  (dt, ()) <- time $ display0 mR
  logStats m "display" dt
  return ()

computeJobs :: IORef Mandulia -> (Julia -> Double) -> IO [Quad]
computeJobs mR curScore = do
  m <- readIORef mR
  i <- readIORef (iface m)
  (js, qs) <- quads mR
  cs <- sortIO curScore . filter (notCached (textures m)) $ js
  let job j = JuliaJob{ jCoords = j, jDoneAction = Nothing }
      js' = map job . take (iJobs i) $ cs
      visible q = not ( quadX q + 2 * quadR q < 0
                      || fromIntegral (width m)  < quadX q - 2 * quadR q
                      || quadY q + 2 * quadR q < 0
                      || fromIntegral (height m) < quadY q - 2 * quadR q
                      ) && quadR q > 0
  reprioritise (jobs m) (const js')
  return $ filter visible qs

black :: Color4 GLclampf
black = Color4 0 0 0 1

drawQuads :: [Quad] -> IO ()
drawQuads qs = do
  clearColor $= black
  clear [ColorBuffer]
  blend $= Enabled
  blendFunc $= (SrcAlpha, OneMinusSrcAlpha)
  texture Texture2D $= Enabled
  mapM_ drawQuad qs
  textureBinding Texture2D $= Nothing
  texture Texture2D $= Disabled
  blend $= Disabled

completeJobs :: IORef Mandulia -> (Julia -> Double) -> IO ()
completeJobs mR curScore = do
  m <- readIORef mR
  tc <- cacheTextures curScore (textures m) . catMaybes . map jDoneAction =<< ↙
      ↳ completed (jobs m)
  writeIORef mR m{ textures = tc }

record :: IORef Mandulia -> IO ()
record mR = do
  m <- readIORef mR
  i <- readIORef (iface m)
  when (iRecord i) $ do
    hSnapshot stdout (Position 0 0) (Size (fromIntegral (width m)) (fromIntegral↙
        ↳ (height m)))

fullscreen :: IORef Mandulia -> Bool -> IO ()
fullscreen mR fs = do
  m <- readIORef mR
  when (fullScreen m /= fs) $ do
    if fs
     then do
      writeIORef mR m{ oldWidth = width m, oldHeight = height m, fullScreen = fs }
      G.fullScreen
     else do
      writeIORef mR m{ fullScreen = fs }
      windowSize $= Size (fromIntegral $ oldWidth m) (fromIntegral $ oldHeight m)

reshape :: IORef Mandulia -> Size -> IO ()
reshape mR (Size w h) = do
  modifyIORef mR $ \m' -> m'{ width = fromIntegral w, height = fromIntegral h }
  m <- readIORef mR
  I.reshape (iface m) (fromIntegral w) (fromIntegral h)
  i <- readIORef (iface m)
  let (x, y, z) = iView i
      s = ceiling $ 2 * iDisplaySize i
  modifyIORef mR $ \m' -> m'{ view = mkView (width m') (height m') x y z }
  viewport $= (Position (-s) (-s), (Size (w + 2 * fromIntegral s) (h + 2 * fromIntegral s)))
  matrixMode $= Projection
  loadIdentity
  ortho (-fromIntegral s) (fromIntegral w + fromIntegral s) (-fromIntegral s) (fromIntegral h + fromIntegral s) (-1) 1
  matrixMode $= Modelview 0
  loadIdentity
  postRedisplay Nothing

timer :: IORef Mandulia -> IO ()
timer mR = do
  m <- readIORef mR
  i <- readIORef (iface m)
  let mspf = ceiling $ 1000 / (iFPS i `max` 0.01)
  addTimerCallback mspf $ timer mR
  postRedisplay Nothing

kmouse :: IORef Mandulia -> Key -> KeyState -> Modifiers -> Position -> IO  ()
kmouse mR (Char     '\27') Down _m _p = key mR "Escape"
kmouse mR (Char         k) Down _m _p = key mR [ k ]
kmouse mR (SpecialKey sk) Down _m _p = case show sk of
                                          'K':'e':'y':key' -> key mR key'
                                          key'             -> key mR key'
kmouse _r _k               _s   _m _p = return () -- FIXME handle everything

key :: IORef Mandulia -> String -> IO ()
key mR k = do
  m <- readIORef mR
  I.keyboard (iface m) k
```

## 26 src/PriorityCache.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module PriorityCache (PriorityCache(..), priorityCache, cache) where

import Sort

data PriorityCache c =
  PriorityCache
    { cSize :: Int
    , cContents :: [c]
    }

priorityCache :: Int -> PriorityCache c
priorityCache size = PriorityCache{ cSize = size, cContents = [] }

cache :: (c -> Double) -> [c] -> PriorityCache c -> IO (PriorityCache c, [c])
cache p xs c = do
  cs <- sortIO p (xs ++ cContents c)
  let (ys, zs) = splitAt (cSize c) cs
  return (c{ cContents = ys }, zs)
```

## 27 src/qsort.c

```
/*
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
*/

#include <stdlib.h>

```

```
#include ”qsort.h”

static int scmp(const void *x, const void *y) {
  const struct sortee *sx = x;
  const struct sortee *sy = y;
  double kx = sx->key;
  double ky = sy->key;
  if (kx < ky) return -1;
  if (kx > ky) return  1;
  return 0;
}

void sort(struct sortee *p, int n) {
  qsort(p, n, sizeof(struct sortee), scmp);
}
```

## 28 src/qsort.h

```
/*
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef SORT_H
#define SORT_H 1

struct sortee {
  double key;
  int value;
};

void sort(struct sortee *p, int n);

#endif
```

## 29 src/ResourcePool.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module ResourcePool(ResourcePool(), resourcePool, acquire, release, withResource) where

import Control.Concurrent.MVar
import Data.Maybe (maybeToList)

type ResourcePool r = MVar (ResourcePool' r)

data ResourcePool' r =
  ResourcePool'
    { rAlloc      :: IO r
    , rBound      :: Int
    , rCount      :: Int
    , rResources  :: [r]
    , rNext       :: MVar r
    }

resourcePool :: IO r -> Int -> IO (ResourcePool r)
resourcePool alloc bound = do
  nr <- newEmptyMVar
  newMVar ResourcePool'
    { rAlloc = alloc
    , rBound = bound
    , rCount = 0
    , rResources = []
    , rNext = nr
    }

acquire :: ResourcePool r -> IO r
acquire p = do
  r <- takeMVar p
  next <- tryTakeMVar $ rNext r
  case next of
    Nothing ->
      case rResources r of
        [] -> if rCount r < rBound r
                then do
                  nr <- rAlloc r
                  putMVar p r{ rCount = rCount r + 1 }
                  return nr
                else do
                  putMVar p r
                  takeMVar $ rNext r
        (nr:rs) ->  do
              putMVar p r{ rResources = rs }
```

```
          return nr
    Just nr ->      do
          putMVar p r
          return nr

release :: ResourcePool r -> r -> IO ()
release p x = do
  r <- takeMVar p
  r0 <- tryTakeMVar $ rNext r
  let rs = maybeToList r0 ++ rResources r
  putMVar p r{ rResources = rs }
  putMVar (rNext r) x

withResource :: ResourcePool r -> (r -> IO b) -> IO b
withResource p action = do
  r <- acquire p
  b <- action r
  release p r
  return b
```

## 30 src/rjulia.c

```
/*
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
*/

#include <math.h>
#include <stdlib.h>

#include "rjulia.h"

#ifndef PRECISION
#define PRECISION 1
#endif

#if PRECISION == 1
typedef float R;
#define cos cosf
#define sqrt sqrtf
#define log logf
#define log2 log2f
#define fmin fminf
#define fmax fmaxf
```

```
#define SCAN "%f"
#else
typedef double R;
#define SCAN "%lf"
#endif

static const int channels = 4;
static const R escapeRadius = 65536.0;
static const R escapeRadius2 = 65536.0 * 65536.0;
static const int escapeLimit = 64;

static const R sqrt5 = 2.23606797749979;
static const R sqrt6 = 2.449489742783178;
static const R sqrt7 = 2.6457513110645907;

struct point {
  unsigned char *pixel;
  R x;
  R y;
};

struct context {
  struct point *points;
  int width;
  int height;
};

struct context *julia_new(int width, int height) {
  if (width > 0 && height > 0) {
    struct context *ctx = calloc(1, sizeof(struct context));
    if (ctx) {
      ctx->points = calloc(width * height, sizeof(struct point));
      if (ctx->points) {
        ctx->width = width;
        ctx->height = height;
        return ctx;
      }
      free(ctx);
    }
  }
  return 0;
}

/*
void julia_delete(struct context *ctx) {
  if (ctx) {
    if (ctx->points) {
      free(ctx->points);
    }
    free(ctx);
  }
}
*/

static inline int min(int x, int y) {
  return x < y ? x : y;
}
```

```
static inline int max(int x, int y) {
  return x > y ? x : y;
}

static inline void colour(R v, unsigned char *r, unsigned char *g, unsigned char *b) {
  int rr = 128 * cos(sqrt5 * v) + 128;
  int gg = 128 * cos(sqrt6 * v) + 128;
  int bb = 128 * cos(sqrt7 * v) + 128;
  *r = min(max(rr, 0), 255);
  *g = min(max(gg, 0), 255);
  *b = min(max(bb, 0), 255);
}

static inline double mandelbrot1(double x, double y) {
  R px = 0, py = 0, cx = x, cy = y, px1, py1, px2, py2, pxy, d2;
  for (int i = 0; i < 16384; ++i) {
    // p_n+1 := p_n * p_n + c
    // d_n+1 := 2 * p_n * d_n + 1
    px2 = px * px;
    py2 = py * py;
    d2 = px2 + py2;
    if (d2 > escapeRadius2) {
      return i - log2(log2(d2)/log2(escapeRadius2));
    }
    pxy = px * py;
    px1 = px2 - py2 + cx;
    py1 = 2 * pxy + cy;
    px = px1;
    py = py1;
  }
  return -1;
}

void julia(struct context *ctx, unsigned char *image, double dcx, double dcy) {
  if (! ctx || ! image) {
    return;
  }
  struct point *points = ctx->points;
  const int width = ctx->width;
  const int height = ctx->height;
  const int stride = width * channels;
  const R cx = dcx;
  const R cy = dcy;
  const R k = 4;
  const R hw = height * k / width;
  const R wh = width  * k / height;
  const R sx = width < height ? k : wh;
  const R sy = width < height ? hw : k;
  const int ntotal = width * height;
  int npoints;
  { // initialize
    npoints = ntotal;
    struct point *p = points;
    for (int j = 0; j < height; ++j) {
      R y = j * sy / height - sy/2;
      for (int i = 0; i < width; ++i) {
        R x = i * sx / width - sx/2;
        p->pixel = image + j * stride + i * channels;
        p->x = x;
        p->y = y;
        p->pixel[0] = 0;
        p->pixel[1] = 0;
        p->pixel[2] = 0;
        p->pixel[3] = 255;
        ++p;
      }
    }
  }
  int ndone = 0;
  int n = 0;
  int escapes;
  R u = mandelbrot1(dcx, dcy);
  u = u > 0 ? 8 * log2(u) : 0;
  do {
    escapes = 0;
    { // iterate
      struct point *p = points;
      for (int k = 0; k < npoints; ++k) {
        R x = p->x;
        R y = p->y;
        int ok = 1;
        for (int e = 0; ok && (e < escapeLimit); ++e) {
          R zx2 = x * x - y * y;
          R zy2 = 2 * x * y;
          R zx = zx2 + cx;
          R zy = zy2 + cy;
          R zd2 = zx * zx + zy * zy;
          if (zd2 >= escapeRadius2) {
            R d = sqrt(zd2);
            // colourify pixel
            unsigned char *pixel = p->pixel;
            R v = (n+e) - log2(log2(d)/log2(escapeRadius));
            v /= 32;
            colour(u + v, &pixel[0], &pixel[1], &pixel[2]);
            R a = fmin(fmax(8 * (v - 0.1) * v, 0), 1);
            pixel[3] = a * 255;
            p->pixel = 0;
            ++escapes;
            ok = 0;
          } else {
            x = zx;
            y = zy;
          }
        }
        p->x = x;
        p->y = y;
        ++p;
      }
      n += escapeLimit;
    }
    { // compact memory
      struct point *src = points;
      struct point *dst = points;
      int ncopied = 0;
      for (int k = 0; k < npoints; ++k) {
        if (! (src->pixel)) {
          ++src;
          ++ndone;
        } else {
          unsigned char *pixel = src->pixel;
          R x = src->x;
          R y = src->y;
          dst->pixel = pixel;
          dst->x = x;
          dst->y = y;
          ++src;
          ++dst;
          ++ncopied;
        }
      }
      npoints = ncopied;
    }
  } while(escapes);
}
```

## 31 src/rjulia.h

```
/*
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef RJULIA_H
#define RJULIA_H 1

struct context;

struct context *julia_new(int width, int height);
/*
void julia_delete(struct context *ctx);
*/
void julia(struct context *ctx, unsigned char *image, double dcx, double dcy);

#endif
```

## 32 src/rts.c

```c
/*
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
*/

char *ghc_rts_opts = "-M50M"; /* we can run in constant space */
```

## 33 src/Snapshot.hs

```haskell
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module Snapshot (hSnapshot, writeSnapshot, snapshotWith) where

import Control.Monad(forM_)
import System.IO(Handle())
import Graphics.UI.GLUT(
  readPixels,
  Position,
  Size(Size),
  PixelData(PixelData),
  PixelFormat(RGB),
  DataType(UnsignedByte))
import Foreign.Marshal.Alloc(allocaBytes)
import Foreign.Ptr(plusPtr)
import qualified Data.ByteString.Internal as BSI
import qualified Data.ByteString as BS

-- save a screenshot to a handle as binary PPM
snapshotWith :: (BS.ByteString -> IO b) -> Position -> Size -> IO b
snapshotWith f p0 vp@(Size vw vh) = do
  let fi q = fromIntegral q
      p6 = "P6\n" ++ show vw ++ " " ++ show vh ++ " 255\n"
  allocaBytes (fi (vw*vh*3)) $ \ptr -> do
    readPixels p0 vp $ PixelData RGB UnsignedByte ptr
    px <- BSI.create (fi $ vw * vh * 3) $ \d -> forM_ [0..vh-1] $ \y ->
      BSI.memcpy
        (d`plusPtr`fi(y*vw*3))
        (ptr`plusPtr`fi ((vh-1-y)*vw*3))
        (fi(vw*3))
    f $ BS.pack (map (toEnum . fromEnum) p6) `BS.append` px

hSnapshot :: Handle -> Position -> Size -> IO ()
hSnapshot h = snapshotWith (BS.hPutStr h)

writeSnapshot :: FilePath -> Position -> Size -> IO ()
writeSnapshot f = snapshotWith (BS.writeFile f)
```

## 34 src/Sort.hsc

```
{-# LANGUAGE ForeignFunctionInterface #-}
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010 Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program. If not, see <http://www.gnu.org/licenses/>.
-}

module Sort where

import Data.Array (array, (!))
import Foreign.Storable
import Foreign.Ptr (Ptr, castPtr)
import Foreign.C.Types (CInt(..), CDouble(..))
import Foreign.Marshal.Array (withArray, peekArray)
import Unsafe.Coerce (unsafeCoerce)

#include "qsort.h"

foreign import ccall safe "qsort.h sort" c_sort :: Ptr () -> CInt -> IO ()

data Sortee = Sortee{ sKey :: !Double, sValue :: !Int }
```

```
instance Storable Sortee where
  sizeOf _ = (#size struct sortee)
  alignment _ = alignment (error "alignment" :: CInt)
  peek ptr = do
    key   <- (#peek struct sortee, key  ) ptr :: IO CDouble
    value <- (#peek struct sortee, value) ptr :: IO CInt
    return Sortee{ sKey = unsafeCoerce key, sValue = fromIntegral value }
  poke ptr (Sortee{ sKey = key, sValue = value }) = do
    (#poke struct sortee, key  ) ptr (unsafeCoerce key :: CDouble)
    (#poke struct sortee, value) ptr (fromIntegral value :: CInt)

sortIO :: (a -> Double) -> [a] -> IO [a]
sortIO p xs = do -- return (sortOn p xs)
  let n  = length xs
      is = [0 .. n - 1]
      ps = map p xs
      a  = array (0, n - 1) (zip is xs)
      ss = zipWith Sortee ps is
  ss' <- withArray ss $ \ptr -> do
    c_sort (castPtr ptr) (fromIntegral n)
    peekArray n ptr
  let is' = map sValue ss'
  return $ map (a !) is'
```

## 35 src/StatsLogger.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module StatsLogger(statsLogger, Logger, Statistics, time) where

import Prelude hiding (map)
import Data.Map (empty, insertWith', map, toAscList)
import Control.Concurrent.MVar (newMVar, modifyMVar_, readMVar)
import Data.Time (getCurrentTime, diffUTCTime)

type Logger = String -> Double -> IO ()
type Statistics = IO [(String, (Double, Double, Double))]

data Stats =
  Stats
```

```
    { count  :: !Double
    , sumX   :: !Double
    , sumXX  :: !Double
    }

stat :: Double -> Stats
stat x = Stats{ count = 1, sumX = x, sumXX = x * x }

combine :: Stats -> Stats -> Stats
combine s t =
  Stats
    { count = count s + count t
    , sumX  = sumX  s + sumX  t
    , sumXX = sumXX s + sumXX t
    }

stats :: Stats -> (Double, Double, Double)
stats s =
  let mean   = sumX  s / count s
      mean2  = sumXX s / count s
      stddev = sqrt $ mean2 - mean * mean
  in (count s, mean, stddev)

statsLogger :: IO (Logger, Statistics)
statsLogger = do
  v <- newMVar empty
  let logStats name value = modifyMVar_ v $ \m -> return $! (insertWith' combine name (stat value) $! m)
      getStats = readMVar v >>= return . toAscList . map stats
  return (logStats, getStats)

time :: IO a -> IO (Double, a)
time x = do
  t0 <- getCurrentTime
  r <- x
  t1 <- getCurrentTime
  return (realToFrac (diffUTCTime t1 t0), r)
```

## 36 src/TextureCache.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}
```

```haskell
module TextureCache (TextureCache(), textureCache, lookupTexture, notCached, cacheTextures) where

import Prelude hiding (lookup)
import Control.Monad (forM, forM_)
import Data.Map (Map, empty, lookup, fromList, union, difference, notMember)
import Data.Either (partitionEithers)

import Graphics.UI.GLUT (TextureObject(), deleteObjectNames)

import Julia
import PriorityCache

type Cachee = Either (Julia, TextureObject) (Julia, (IO TextureObject, IO()))

data TextureCache =
  TextureCache
    { tcCache :: PriorityCache Cachee
    , tcMap   :: Map Julia TextureObject
    }

textureCache :: Int -> TextureCache
textureCache size = TextureCache{ tcCache = priorityCache size, tcMap = empty }

lookupTexture :: TextureCache -> Julia -> Maybe TextureObject
lookupTexture tc j = j `lookup` tcMap tc

notCached :: TextureCache -> Julia -> Bool
notCached tc j = j `notMember` tcMap tc

cacheTextures :: (Julia -> Double) -> TextureCache -> [Cachee] -> IO TextureCache
cacheTextures jscore tc news = do
  (pc,   olds ) <- cache (either (jscore . fst) (jscore . fst)) news (tcCache tc )
  let (texs, toups) = partitionEithers (cContents pc)
      (dels, noups) = partitionEithers olds
  deleteObjectNames (map snd dels)
  forM_ noups $ \(_j, (_up, no)) -> no
  upped <- forM toups $ \(j, (up, _no)) -> do
    t <- up
    return (j, t)
  return tc
    { tcCache = pc{ cContents = map Left (upped ++ texs) }
    , tcMap = (tcMap tc `difference` fromList dels) `union` fromList upped
    }
```

## 37 src/Utils.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module Utils (phi, phi', clamp, modifyIORef') where

import Data.IORef(IORef(), readIORef, writeIORef)

phi  :: Double
phi  = (sqrt 5 + 1) / 2

phi' :: Double
phi' = (sqrt 5 - 1) / 2

clamp :: Ord r => r -> r -> r -> r
clamp mi ma x = ma `min` x `max` mi

modifyIORef' :: IORef a -> (a -> a) -> IO ()
modifyIORef' r m = readIORef r >>= \v -> writeIORef r $! m v
```

## 38 src/Vector.hs

```
{-
Mandulia -- Mandelbrot/Julia explorer
Copyright (C) 2010  Claude Heiland-Allen <claude@mathr.co.uk>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
-}

module Vector
  ( R
  , V(..)
  , M(..)
  , (^*)
  , (^/)
  , (^+^)
  , (^-^)
  , (^|-|^)
  , (^^*^)
  , (^^*^^)
  , translate
  , scale
  , rotate
  ) where

type R = Double
data V = V !R !R !R deriving (Show, Read, Eq, Ord)
data M = M !V !V !V deriving (Show, Read, Eq, Ord)

translate :: R -> R -> M
translate x y = M (V 1 0 x) (V 0 1 y) (V 0 0 1)

scale :: R -> R -> M
scale x y = M (V x 0 0) (V 0 y 0) (V 0 0 1)

rotate :: R -> M
rotate a = M (V c s 0) (V (-s) c 0) (V 0 0 1)
  where
    s = sin a
    c = cos a

(^*) :: V -> R -> V
(V a b c)  ^*   x = V (a*x) (b*x) (c*x)

(^/) :: V -> R -> V
(V a b c)  ^/   x = V (a/x) (b/x) (c/x)

(^+^) :: V -> V -> V
(V a b c)  ^+^  (V x y z) = V (a+x) (b+y) (c+z)

(^-^) :: V -> V -> V
(V a b c)  ^-^  (V x y z) = V (a-x) (b-y) (c-z)

dot :: V -> V -> R
(V a b c) `dot` (V x y z) = a*x + b*y + c*z

(^^*^) :: M -> V -> V
(M (V a b c) (V d e f) (V g h i)) ^^*^ (V r u x) =
  V (a * r + b * u + c * x)
    (d * r + e * u + f * x)
    (g * r + h * u + i * x)

(^^*^^) :: M -> M -> M
(M (V a b c) (V d e f) (V g h i)) ^^*^^ (M (V r s t) (V u v w) (V x y z)) =
  M (V (a * r + b * u + c * x) (a * s + b * v + c * y) (a * t + b * w + c * z))
    (V (d * r + e * u + f * x) (d * s + e * v + f * y) (d * t + e * w + f * z))
    (V (g * r + h * u + i * x) (g * s + h * v + i * y) (g * t + h * w + i * z))

(^|-|^) :: V -> V -> R
u ^|-|^ v = let d = u ^-^ v in sqrt $ d `dot` d
```

## 39 TODO

TODO

```
Bugs To Fix
-----------

Problem:  running out of bugs to fix.


Features To Enhance
-------------------

Feature:  script interface should include level to radius fading.
Feature:  script interface should include view validity testing.
Feature:  script interface should include more complete input handling.
Feature:  'transition.lua' should work without 'dt' depending on 'z'.
Feature:  large tiled texture might be more efficient when drawing.


Other Known Issues
------------------

Problem:  script errors terminate 'mandulia' abruptly.
Cause:    lack of error checking in 'Interface.hs'.
Solution: don't write buggy scripts.

Problem:  real-time operation lags Julia rendering at rapid movements.
Cause:    Your system is too SLOW to play this!
Solution: get a faster computer.

Problem:  recording enabled leads to higher quality.
Cause:    main thread blocks on image output, Julia renderer threads don't.
Solution: use another worker thread for image output (but HQ recs are nice).

Problem:  scripts printing to 'stdout' corrupts recordings.
Cause:    recordings go to 'stdout', which expects nothing but recordings.
Solution: somehow redirect all Lua 'stdout' to 'stderr'.

Problem:  script's 'render()' doesn't get called when the window is hidden.
Cause:    that's exactly the way GLUT is supposed to behave.
Solution: switch to a less smelly UI library to wrap OpenGL.  SDL?

Problem:  'extra/' in git repository contains unrelated code.
Cause:    ppmtoy4m was a video encoding bottleneck, needed speeding up.
Solution: move that code to a different repository.

Problem:  very occasional glibc realloc() crashes.
Cause:    possibly an nVidia driver issue?
Solution: unknown. seems sporadic and random.
Cause:    possibly a Lua stack overflow?
Solution: fixed in eb8a82052c7f50a64bd23d19278757a649f23f9c
```