

fractal-channel-hopping

Claude Heiland-Allen

2011–2019

Contents

1	audio/fch.pd	2
2	audio/voice~.pd	4
3	COPYING	5
4	.gitignore	17
5	NEWS	17
6	photo/fractal-photo-mosaic.c	18
7	photo/Makefile	25
8	photo/prepare.sh	25
9	photo/README.md	25
10	README	26
11	src/channel.c	26
12	src/channel.h	28
13	src/config.h	29
14	src/fch.frag	30
15	src/list.c	31
16	src/list.h	33
17	src/main.c	34
18	src/Makefile	45
19	src/pfifo.c	46
20	src/pfifo.h	47
21	src/record.c	48
22	src/record.h	50
23	src/s2c.sh	51
24	src/yuv2rgb.frag	51
25	start.sh	52

1 audio/fch.pd

```

#N canvas 3 63 450 524 10;
#X obj 21 20 table \${0}-notes 24;
#X obj 21 42 loadbang;
#X obj 21 86 s \${0}-notes;
5 #X obj 21 108 loadbang;
#X msg 21 151 1;
#X obj 75 197 + 1;
#X obj 75 219 mod 24;
#X obj 21 195 f;
10 #X obj 21 217 t f f;
#X obj 21 239 mod 4;
#X obj 21 261 pack f f;
#X obj 72 241 tabread \${0}-notes;
#X obj 21 283 route 0 1 2 3;

```

```

15  #X obj 18 464 dac~;
    #X obj 277 125 loadbang;
    #X msg 277 147 \; pd dsp 1;
    #X obj 21 313 voice~;
    #X obj 31 333 voice~;
20  #X obj 39 355 voice~;
    #X obj 51 373 voice~;
    #X obj 18 396 hip~ 10;
    #X obj 18 420 lop~ 10000;
    #X obj 131 283 route 0 1 2 3;
25  #X obj 131 313 voice~;
    #X obj 141 333 voice~;
    #X obj 149 355 voice~;
    #X obj 161 373 voice~;
    #X obj 128 396 hip~ 10;
30  #X obj 128 418 lop~ 10000;
    #X msg 21 64 0 0 9 5 2 10 7 3 0 8 5 1 10 6 3 11 8 4 1 9 6 2 11 7 4
        ;
    #X obj 21 173 metro 2500;
    #X obj 208 478 writesf~ 2;
35  #X msg 262 260 open -bytes 4 fch.wav \, start;
    #X obj 21 130 delay 1000;
    #X obj 280 289 delay 1.8e+06;
    #X msg 280 345 stop;
    #X msg 280 394 \; pd quit;
40  #X obj 293 370 delay 1000;
    #X obj 262 211 delay 60000;
    #X floatatom 154 213 5 0 0 0 - - -, f 5;
    #X connect 1 0 29 0;
    #X connect 3 0 33 0;
45  #X connect 4 0 30 0;
    #X connect 5 0 6 0;
    #X connect 6 0 7 1;
    #X connect 7 0 5 0;
    #X connect 7 0 8 0;
50  #X connect 7 0 39 0;
    #X connect 8 0 9 0;
    #X connect 8 1 11 0;
    #X connect 9 0 10 0;
    #X connect 10 0 12 0;
55  #X connect 10 0 22 0;
    #X connect 11 0 10 1;
    #X connect 12 0 16 0;
    #X connect 12 1 17 0;
    #X connect 12 2 18 0;
60  #X connect 12 3 19 0;
    #X connect 14 0 15 0;
    #X connect 16 0 20 0;
    #X connect 17 0 20 0;
    #X connect 18 0 20 0;
65  #X connect 19 0 20 0;
    #X connect 20 0 21 0;
    #X connect 21 0 13 0;
    #X connect 21 0 31 0;
    #X connect 22 0 23 0;
70  #X connect 22 1 24 0;
    #X connect 22 2 25 0;

```

```

#X connect 22 3 26 0;
#X connect 23 0 27 0;
#X connect 24 0 27 0;
75 #X connect 25 0 27 0;
#X connect 26 0 27 0;
#X connect 27 0 28 0;
#X connect 28 0 13 1;
#X connect 28 0 31 1;
80 #X connect 29 0 2 0;
#X connect 30 0 7 0;
#X connect 32 0 31 0;
#X connect 33 0 4 0;
#X connect 33 0 38 0;
85 #X connect 34 0 35 0;
#X connect 34 0 37 0;
#X connect 35 0 31 0;
#X connect 37 0 36 0;
#X connect 38 0 32 0;
90 #X connect 38 0 34 0;

```

2 audio/voice~.pd

```

#N canvas 3 58 450 300 10;
#X obj 174 19 inlet;
#X obj 174 41 mtof;
#X obj 30 33 noise~;
5 #X obj 83 64 vcf~ 50;
#X obj 19 286 outlet~;
#X obj 83 84 vcf~ 50;
#X obj 83 104 vcf~ 50;
#X obj 83 124 vcf~ 50;
10 #X obj 83 144 vcf~ 50;
#X obj 83 164 vcf~ 50;
#X obj 83 184 vcf~ 50;
#X obj 83 205 vcf~ 50;
#X obj 83 225 vcf~ 50;
15 #X obj 83 245 vcf~ 50;
#X obj 83 265 vcf~ 50;
#X obj 174 63 * 2;
#X obj 174 85 * 2;
#X obj 174 107 * 2;
20 #X obj 174 129 * 2;
#X obj 174 151 * 2;
#X obj 174 173 * 2;
#X obj 174 195 * 2;
#X obj 174 217 * 2;
25 #X obj 174 239 * 2;
#X obj 174 261 * 2;
#X connect 0 0 1 0;
#X connect 1 0 3 1;
#X connect 1 0 15 0;
30 #X connect 2 0 3 0;
#X connect 2 0 5 0;
#X connect 2 0 6 0;
#X connect 2 0 7 0;
#X connect 2 0 8 0;
35 #X connect 2 0 9 0;

```

```

#X connect 2 0 10 0;
#X connect 2 0 11 0;
#X connect 2 0 12 0;
#X connect 2 0 13 0;
40 #X connect 2 0 14 0;
#X connect 3 0 4 0;
#X connect 5 0 4 0;
#X connect 6 0 4 0;
#X connect 7 0 4 0;
45 #X connect 8 0 4 0;
#X connect 9 0 4 0;
#X connect 10 0 4 0;
#X connect 11 0 4 0;
#X connect 12 0 4 0;
50 #X connect 13 0 4 0;
#X connect 14 0 4 0;
#X connect 15 0 5 1;
#X connect 15 0 16 0;
#X connect 16 0 6 1;
55 #X connect 16 0 17 0;
#X connect 17 0 7 1;
#X connect 17 0 18 0;
#X connect 18 0 8 1;
#X connect 18 0 19 0;
60 #X connect 19 0 9 1;
#X connect 19 0 20 0;
#X connect 20 0 10 1;
#X connect 20 0 21 0;
#X connect 21 0 11 1;
65 #X connect 21 0 22 0;
#X connect 22 0 12 1;
#X connect 22 0 23 0;
#X connect 23 0 13 1;
#X connect 23 0 24 0;
70 #X connect 24 0 14 1;

```

3 COPYING

GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007

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

Preamble

10 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,
15 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

20 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
25 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
30 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
35 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.

40 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
45 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.

50 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
55 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.

60 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
65 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
70 modification follow.

TERMS AND CONDITIONS

0. Definitions.

75 "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.

80 "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.

85 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.

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

95 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.

100 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.

105 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.

115 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.

120 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.

125 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.

195 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;
200 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.

205 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.

210 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:

215 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
220 "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
225 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.

230 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.

235 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
240 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.

245 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

305 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.

310 "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
315 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
320 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
325 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
330 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
335 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
340 source code form), and must require no special password or key for
unpacking, reading or copying.

7. Additional Terms.

345 "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
350 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
355 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.

360 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:

- 365 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
- 370 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
- 375 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
- 380 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
- 385 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

390 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

395 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

400 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;

405 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

410 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).

415 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.

590 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,
595 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.

600 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
605 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
610 SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided
615 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.

620 END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

625 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
630 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.

635 <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
640 (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
645 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/>>.

650 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:

655 <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.

660 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".

665 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/>>.

670 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>>.

4 .gitignore

```
src/fractal-channel-hopping
src/*.frag.c
photo/fractal-photo-mosaic
get-iplayer
5  rtmpdump
   tmp
   *.ogv
   *.m2v
   *.wav
10 *.mp2
   *.mkv
   *.data
   *.mp4
   *.mpeg
15 *.webm
```

5 NEWS

```
v6      play videos from folders
v5      fix for get-iplayer mode name changes
v4      mplayer simplified stuff
v3      graphics speed boost etc
5  v2      preliminary sound mixing
   v1      first publicised version
```

6 photo/fractal-photo-mosaic.c

```

#include <assert.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
5  #include <GL/glew.h>
#include <GLFW/glfw3.h>

#define COUNT (103+1)
#define WIDTH 1920
10 #define HEIGHT 1080

static unsigned char raw[COUNT][HEIGHT][WIDTH][3];
static unsigned char raw16[COUNT][16][16][3];

15 static double lab16[COUNT][16][16][3];
static double lab2[COUNT][2][2][3];

static float graph[COUNT][8][8];

20 static int visited[COUNT];

static unsigned char out[HEIGHT][WIDTH][3];

static const char *vert =
25 "#version 400 core\n"
"uniform vec2 delta;\n"
"uniform float zoom;\n"
"layout(location = 0) in vec2 pos;\n"
"layout(location = 1) in vec2 tc;\n"
30 "out vec2 c;\n"
"void main() {\n"
"    gl_Position = vec4(zoom * (pos - delta) + delta, 0.0, 1.0);\n"
"    c = vec2(tc.x, 1.0 - tc.y);\n"
"}\n"
35 ;

static const char *frag =
"#version 400 core\n"
"uniform sampler2DArray tex;\n"
40 "uniform sampler2DArray map;\n"
"uniform int ix;\n"
"uniform vec3 blend;\n"
"in vec2 c;\n"
"layout(location = 0) out vec3 colour;\n"
45 "void main() {\n"
"    float l = textureQueryLod(tex, c).y;\n"
"    vec3 p0 = vec3(c, float(ix));\n"
"    vec3 p1 = vec3(8.0 * p0.xy, texture(map, p0).x);\n"
"    p1.xy -= floor(p1.xy);\n"
50 "    vec3 p2 = vec3(8.0 * p1.xy, texture(map, p1).x);\n"
"    p2.xy -= floor(p2.xy);\n"
"    vec3 sum = vec3(0.0);\n"
"    sum += textureLod(tex, p0, l).rgb * blend.x;\n"
"    sum += textureLod(tex, p1, l + 3).rgb * blend.y;\n"
55 "    sum += textureLod(tex, p2, l + 6).rgb * blend.z;\n"

```

```

"    colour = sum;\n"
"}\n"
;

60 static void debug_program(GLuint program, const char *name) {
    if (program) {
        GLint linked = GL_FALSE;
        glGetProgramiv(program, GL_LINK_STATUS, &linked);
        if (linked != GL_TRUE) {
65             fprintf(stderr, "%s: OpenGL shader program link failed\n", name);
        }
        GLint length = 0;
        glGetProgramiv(program, GL_INFO_LOG_LENGTH, &length);
        char *buffer = (char *) malloc(length + 1);
70         glGetProgramInfoLog(program, length, 0, buffer);
        buffer[length] = 0;
        if (buffer[0]) {
            fprintf(stderr, "%s: OpenGL shader program info log\n", name);
            fprintf(stderr, "%s\n", buffer);
75         }
        free(buffer);
        assert(linked == GL_TRUE);
    } else {
        fprintf(stderr, "%s: OpenGL shader program creation failed\n", name);
80     }
}

static void debug_shader(GLuint shader, GLenum type, const char *name) {
    const char *tname = 0;
85     switch (type) {
        case GL_VERTEX_SHADER:    tname = "vertex";    break;
        case GL_FRAGMENT_SHADER:  tname = "fragment";  break;
        case GL_COMPUTE_SHADER:    tname = "compute";   break;
        default:                  tname = "unknown";   break;
90     }
    if (shader) {
        GLint compiled = GL_FALSE;
        glGetShaderiv(shader, GL_COMPILE_STATUS, &compiled);
        if (compiled != GL_TRUE) {
95             fprintf(stderr, "%s: OpenGL %s shader compile failed\n", name, tname);
        }
        GLint length = 0;
        glGetShaderiv(shader, GL_INFO_LOG_LENGTH, &length);
        char *buffer = (char *) malloc(length + 1);
100        glGetShaderInfoLog(shader, length, 0, buffer);
        buffer[length] = 0;
        if (buffer[0]) {
            fprintf(stderr, "%s: OpenGL %s shader info log\n", name, tname);
            fprintf(stderr, "%s\n", buffer);
105        }
        free(buffer);
        assert(compiled == GL_TRUE);
    } else {
        fprintf(stderr, "%s: OpenGL %s shader creation failed\n", name, tname);
110    }
}

```

```

static void compile_shader(GLint program, GLenum type, const char *name, const ↵
    ↵ GLchar *source) {
    GLuint shader = glCreateShader(type);
115   glShaderSource(shader, 1, &source, 0);
    glCompileShader(shader);
    debug_shader(shader, type, name);
    glAttachShader(program, shader);
    glDeleteShader(shader);
120 }

static GLint compile_program(const char *name, const GLchar *vert, const GLchar ↵
    ↵ *frag) {
    GLint program = glCreateProgram();
    if (vert) { compile_shader(program, GL_VERTEX_SHADER, name, vert); }
125   if (frag) { compile_shader(program, GL_FRAGMENT_SHADER, name, frag); }
    glLinkProgram(program);
    debug_program(program, name);
    return program;
}

130 static double xyz2lab_f(double t)
{
    static const double e = 0.008856;
    static const double k = 903.3;
    if (t > e)
135     return cbrt(t);
    else
        return (k * t + 16) / 116;
}

static void xyz2lab(double x, double y, double z, double *l, double *a, double *↵
    ↵ b)
140 {
    static const double xn = 0.95047;
    static const double yn = 1.00000;
    static const double zn = 1.08883;
    x /= xn;
145   y /= yn;
    z /= zn;
    x = xyz2lab_f(x);
    y = xyz2lab_f(y);
    z = xyz2lab_f(z);
150   *l = 116 * y - 16;
    *a = 500 * (x - y);
    *b = 200 * (y - z);
}

155 static double srgb2xyz_f(double c)
{
    if (c < 0.04045)
        return c / 12.92;
    else
160     return pow((c + 0.055) / 1.055, 2.4);
}

static void srgb2xyz(double r, double g, double b, double *x, double *y, double ↵
    ↵ *z)
{
    static const double m[3][3] =
165     { { 0.4124, 0.3576, 0.1805 }

```

```

        , { 0.2126, 0.7152, 0.0722 }
        , { 0.0193, 0.1192, 0.9505 }
    };
    r = srgb2xyz_f(r);
170    g = srgb2xyz_f(g);
    b = srgb2xyz_f(b);
    *x = m[0][0] * r + m[0][1] * g + m[0][2] * b;
    *y = m[1][0] * r + m[1][1] * g + m[1][2] * b;
    *z = m[2][0] * r + m[2][1] * g + m[2][2] * b;
175 }

static void srgb2lab(double r, double g, double b, double *l, double *a, double *
    ↪ *bb)
{
180    double x, y, z;
    srgb2xyz(r, g, b, &x, &y, &z);
    xyz2lab(x, y, z, l, a, bb);
}

185 extern int main()
{
    srand(0x1cedcafe);

190    FILE *fraw = fopen("image.data", "rb");
    fread(&raw[0][0][0][0], (COUNT-1) * WIDTH * HEIGHT * 3, 1, fraw);
    fclose(fraw);

    fraw = fopen("thumbs.data", "rb");
195    fread(&raw16[0][0][0][0], (COUNT-1) * 16 * 16 * 3, 1, fraw);
    fclose(fraw);

    glfwInit();
    glfwWindowHint(GLFW_CONTEXT_VERSION_MAJOR, 4);
200    glfwWindowHint(GLFW_CONTEXT_VERSION_MINOR, 0);
    glfwWindowHint(GLFW_OPENGL_FORWARD_COMPAT, GL_TRUE);
    glfwWindowHint(GLFW_OPENGL_PROFILE, GLFW_OPENGL_CORE_PROFILE);
    glfwWindowHint(GLFW_OPENGL_DEBUG_CONTEXT, GL_TRUE);
    glfwWindowHint(GLFW_RESIZABLE, GL_FALSE);
205    GLFWwindow *window = glfwCreateWindow(WIDTH, HEIGHT, "fractalize", 0, 0);
    glfwMakeContextCurrent(window);
    if (! window) {
        fprintf(stderr, "glfw\n");
        return 1;
210    }
    glewInit();

    GLuint program = compile_program("shader", vert, frag);
    glUseProgram(program);
215    GLint utex = glGetUniformLocation(program, "tex");
    GLint umap = glGetUniformLocation(program, "map");
    GLint uix = glGetUniformLocation(program, "ix");
    GLint uzoom = glGetUniformLocation(program, "zoom");
    GLint udelta = glGetUniformLocation(program, "delta");
220    GLint ublend = glGetUniformLocation(program, "blend");
    glUniform1i(utex, 0);

```

```

    glUniform1i(umap, 1);

    GLuint vao;
225   glGenVertexArrays(1, &vao);
    glBindVertexArray(vao);
    GLuint vbo;
    glGenBuffers(1, &vbo);
    glBindBuffer(GL_ARRAY_BUFFER, vbo);
230   GLfloat vbo_data[] =
        { -1, -1, 0, 0
          , -1, 1, 0, 1
          , 1, -1, 1, 0
          , 1, 1, 1, 1
235   };
    glBufferData(GL_ARRAY_BUFFER, 16 * sizeof(GLfloat), vbo_data, GL_STATIC_DRAW);
    glVertexAttribPointer(0, 2, GL_FLOAT, GL_FALSE, 4 * sizeof(GLfloat), 0);
    glVertexAttribPointer(1, 2, GL_FLOAT, GL_FALSE, 4 * sizeof(GLfloat), ((char *)↵
        ↵ 0) + 2 * sizeof(GLfloat));
    glEnableVertexAttribArray(0);
240   glEnableVertexAttribArray(1);

    GLuint tex[2];
    glGenTextures(2, &tex[0]);

245   glActiveTexture(GL_TEXTURE0 + 0);
    glBindTexture(GL_TEXTURE_2D_ARRAY, tex[0]);
    glTexImage3D(GL_TEXTURE_2D_ARRAY, 0, GL_RGB, WIDTH, HEIGHT, COUNT, 0, GL_RGB, ↵
        ↵ GL_UNSIGNED_BYTE, &raw[0][0][0][0]);
    glGenerateMipmap(GL_TEXTURE_2D_ARRAY);
    glTexParameteri(GL_TEXTURE_2D_ARRAY, GL_TEXTURE_MIN_FILTER, ↵
        ↵ GL_LINEAR_MIPMAP_LINEAR);
250   glTexParameteri(GL_TEXTURE_2D_ARRAY, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D_ARRAY, GL_TEXTURE_WRAP_S, GL_CLAMP_TO_EDGE);
    glTexParameteri(GL_TEXTURE_2D_ARRAY, GL_TEXTURE_WRAP_T, GL_CLAMP_TO_EDGE);

    for (int k = 0; k < COUNT - 1; ++k)
255   for (int j = 0; j < 16; ++j)
    for (int i = 0; i < 16; ++i)
        srgb2lab(raw16[k][j][i][0]/255.0, raw16[k][j][i][1]/255.0, raw16[k][j][i]↵
            ↵ ][2]/255.0, &lab16[k][j][i][0], &lab16[k][j][i][1], &lab16[k][j][i]↵
            ↵ ][2]);

    for (int k = 0; k < COUNT - 1; ++k)
260   for (int j = 0; j < 2; ++j)
    for (int i = 0; i < 2; ++i)
    {
        double l = 0;
        double a = 0;
265   double b = 0;
        for (int jj = 0; jj < 8; ++jj)
        for (int ii = 0; ii < 8; ++ii)
        {
270   l += lab16[k][8 * j + jj][8 * i + ii][0];
            a += lab16[k][8 * j + jj][8 * i + ii][1];
            b += lab16[k][8 * j + jj][8 * i + ii][2];
        }
        l /= 8 * 8;

```

```

    a /= 8 * 8;
275    b /= 8 * 8;
    lab2[k][j][i][0] = 1;
    lab2[k][j][i][1] = a;
    lab2[k][j][i][2] = b;
}
280
for (int k = 0; k < COUNT - 1; ++k)
for (int j = 0; j < 8; ++j)
for (int i = 0; i < 8; ++i)
{
285    double min_metric = 1.0 / 0.0;
    int min_index = -1;
    for (int kk = 0; kk < COUNT - 1; ++kk)
    {
        if (kk == k) continue;
290        double s = 0;
        for (int jj = 0; jj < 2; ++jj)
        for (int ii = 0; ii < 2; ++ii)
        for (int c = 0; c < 3; ++c)
        {
295            double x = lab16[k][2*j + jj][2*i + ii][c];
            double y = lab2[kk][jj][ii][c];
            double d = x - y;
            s += d * d;
        }
300        if (s < min_metric)
        {
            min_metric = s;
            min_index = kk;
        }
305    }
    graph[k][j][i] = min_index;
}
for (int j = 0; j < 8; ++j)
for (int i = 0; i < 8; ++i)
310    graph[COUNT - 1][j][i] = 0;

glActiveTexture(GL_TEXTURE0 + 1);
glBindTexture(GL_TEXTURE_2D_ARRAY, tex[1]);
glTexImage3D(GL_TEXTURE_2D_ARRAY, 0, GL_R32F, 8, 8, COUNT, 0, GL_RED, GL_FLOAT,
    ↵ , &graph[0][0][0]);
315 glTexParameterf(GL_TEXTURE_2D_ARRAY, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameterf(GL_TEXTURE_2D_ARRAY, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTexParameterf(GL_TEXTURE_2D_ARRAY, GL_TEXTURE_WRAP_S, GL_CLAMP_TO_EDGE);
glTexParameterf(GL_TEXTURE_2D_ARRAY, GL_TEXTURE_WRAP_T, GL_CLAMP_TO_EDGE);

320 int ix = COUNT-1;
int zoomi = 4;
int zoomj = 4;
int speed = 150;
double bdx = 0, bdy = 0;
325 for (int frame = 1; frame <= 15 * 60 * 60; ++frame)
{
    if ((frame % speed) == 0)
    {
        ix = graph[ix][zoomj][zoomi];
    }
}

```

```

330     int mi = 0x7fffffff;
        for (int j = 0; j < 8; ++j)
        for (int i = 0; i < 8; ++i)
        {
            int k = graph[ix][j][i];
335             int m = visited[k];
            mi = m < mi ? m : mi;
        }
        int n = 0;
        for (int j = 0; j < 8; ++j)
        for (int i = 0; i < 8; ++i)
340         {
            int k = graph[ix][j][i];
            int m = visited[k];
            n += mi == m;
345         }
        int coin = rand() % n;
        n = 0;
        for (int j = 0; j < 8; ++j)
        for (int i = 0; i < 8; ++i)
350         {
            int k = graph[ix][j][i];
            int m = visited[k];
            if (mi == m)
            {
355                 if (coin == n)
                    {
                        zoomj = j;
                        zoomi = i;
                        visited[k] += 1;
360                     }
                    n += 1;
            }
        }
    }
}

365 // zoom blending
double k = ((frame % speed) + 0.5) / speed;
double zoom = pow(8, k); // hardcoded power - grid size...
double blend2 = 1 - cos(2 * 3.141592653 * (k + 0) / 3);
double blend1 = 1 - cos(2 * 3.141592653 * (k + 1) / 3);
370 double blend0 = 1 - cos(2 * 3.141592653 * (k + 2) / 3);
double blendt = blend0 + blend1 + blend2;
blend0 /= blendt;
blend1 /= blendt;
blend2 /= blendt;
375 double dx = zoomi * 8 / (8 - 1.0);
double dy = 8 - zoomj * 8 / (8 - 1.0);
dx /= 4;
dy /= 4;
dx -= 1;
380 dy -= 1;
bdx *= 0.95;
bdy *= 0.95;
bdx += 0.05 * dx;
bdy += 0.05 * dy;
385 glUniform1i(uix, ix);
glUniform1f(uzoom, zoom);

```



```

    glUniform2f(udelta, bdx, bdy);
    glUniform3f(ublend, blend0, blend1, blend2);
    glDrawArrays(GL_TRIANGLE_STRIP, 0, 4);
390  glfwSwapBuffers(window);
    glReadPixels(0, 0, WIDTH, HEIGHT, GL_RGB, GL_UNSIGNED_BYTE, &out[0][0][0]);
    printf("P6\n%d %d\n255\n", WIDTH, HEIGHT);
    for (int j = HEIGHT - 1; j >= 0; --j)
    {
395      fwrite(&out[j][0][0], WIDTH * 3, 1, stdout);
    }
    fflush(stdout);
  }
  return 0;
400 }

```

7 photo/Makefile

```

fractal-photo-mosaic: fractal-photo-mosaic.c
    gcc -std=c99 -Wall -Wextra -pedantic -O3 -o fractal-photo-mosaic fractal.c \
    ↪ -photo-mosaic.c -lglfw -lGL -lGLEW -lm

```

8 photo/prepare.sh

```

#!/bin/sh
for i in *.jpeg *.png
do
    convert "${i}" -geometry 1920x1080^ -gravity center ↵
    ↪ -extent 1920x1080 -blur 0x32 "/tmp/
    ↪ /background.png"
5    convert "${i}" -geometry 1920x1080 -gravity center -alpha opaque -background ↵
    ↪ transparent -extent 1920x1080 "/tmp/foreground.png"
    composite -compose Over -gravity center "/tmp/foreground.png" "/tmp/background/
    ↪ .png" "${i}.ppm"
done
for i in *.ppm
do
10    cat "${i}" | tail -c $((1920 * 1080 * 3))
done > image.data
for i in *.ppm
do
    convert "${i}" -geometry '16x16!' "/tmp/thumb.ppm"
15    cat "/tmp/thumb.ppm" | tail -c $((16 * 16 * 3))
done > thumbs.data

```

9 photo/README.md

fractal-photo-mosaic

render a zooming fractal video from a collection of images

5

usage

mkdir images

10

cd images

```

wget *.jpeg *.png # needs exactly 103 images total, TODO FIXME hardcoding
../prepare.sh
cd ..
ln -s images/image.data
15 ln -s images/thumbs.data
make
./fractal-photo-mosaic |
ffmpeg -i soundtrack.wav -f image2pipe -codec ppm -framerate 60 -i - \
    -pix_fmt yuv420p -profile:v high -level:v 4.1 -b:v 20M -b:a 192k \
20 fractal-photo-mosaic.mkv

```

10 README

QUICK START

```

# get a computer with graphics drivers supporting OpenGL/GLSL 130
# on Debian-based systems (you might need multimedia repositories)
5 sudo apt install \
    build-essential \
    libglew1.5-dev \
    freeglut3-dev \
    libjack-jackd2-dev \
10 jackd \
    mjpegtools \
    mplayer \
    ffmpeg
# get the source, if you're reading this you probably already have
15 git clone http://code.mathr.co.uk/fractal-channel-hopping.git
cd fractal-channel-hopping/
git tag -ln99
git checkout v6 # or a different tag
# prepare videos
20 mkdir v
cd v
ln -s /path/to/some/videos/ "Channel 01"
ln -s /path/to/more/videos/ "Channel 02"
# etc, up to 12 channels
25 # start JACK server
../start.sh
# watch tv
# 'Shift-R' to start/stop recording
# 'F11' fullscreen
30 # 'Shift-Q' or 'ESC' to quit

```

NOTES

```

35 be patient during startup (takes a few seconds)
sometimes it doesn't quit cleanly, "killall mplayer" perhaps
tested with NVIDIA proprietary drivers and AMD open source drivers
recording is in PAL DVD format (..mpeg)

```

11 src/channel.c

```

/*
fractal-channel-hopping -- infinite fractal television zoom
Copyright (C) 2011,2015,2019 Claude Heiland-Allen

```

```

5   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.

10  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.

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

#define _DEFAULT_SOURCE

20  #include <pthread.h>
   #include <stdio.h>
   #include <stdlib.h>
   #include <string.h>
25  #include <unistd.h>

   #include "config.h"
   #include "channel.h"

30  void *channel_main(void *arg) {
   struct channel *channel = arg;
   if (channel->width == 256 && channel->height == 256) {
       int bytes = 256 * 256 + 128 * 128 * 2;
       const char *fmt = "mplayer -quiet -really-quiet -loop 0 -shuffle -aspect 1/1 ↵
           ↵ -vf scale=256:256 -ao 'jack:name=%s:port=fch.*%s_[1-2] ' -vo 'yuv4mpeg ↵
           ↵ : file=tmp/%s.fifo ' -fixed-vo '%s/' '*.mp4 >'tmp/%s.mplayer.log ' 2>&1 & ↵
           ↵ cat 'tmp/%s.fifo '"';
35  int cmdlen = strlen(fmt) + 6 * strlen(channel->name) + 64;
       char *cmd = malloc(cmdlen);
       snprintf(cmd, cmdlen - 2, fmt, channel->name, channel->name, channel->name, ↵
           ↵ channel->name, channel->name, channel->name);
       cmd[cmdlen-1] = 0;
       FILE *video;
40  if ((video = popen(cmd, "r"))) {
       const char *vhdr25 = "YUV4MPEG2 W256 H256 F25:1 Ip A1:1\n";
       const char *vhdr50 = "YUV4MPEG2 W256 H256 F50:1 Ip A1:1\n";
       const char *fhdr = "FRAME\n";
       char hdr[64];
45  if (1 != fread(hdr, strlen(vhdr25), 1, video)) { channel->aborted = 1; ↵
           ↵ goto cleanup; }
       hdr[strlen(vhdr25)] = 0;
       if (strcmp(vhdr25, hdr) && strcmp(vhdr50, hdr)) { channel->aborted = 2; ↵
           ↵ goto cleanup; }
       while (! channel->quit) {
           if (1 != fread(hdr, strlen(fhdr), 1, video)) { channel->aborted = 3; ↵
               ↵ break; }
50  hdr[strlen(fhdr)] = 0;
           if (strcmp(fhdr, hdr)) { channel->aborted = 4; break; }
           if (1 != fread(channel->image, bytes, 1, video)) { channel->aborted = 5; ↵
               ↵ break; }

```

```

        }
        cleanup:
55     pclose(video);
    }
    free(cmd);
}
if (channel->aborted) {
60     fprintf(stderr, "channel '%s' aborted '%d'\n", channel->name, channel->
        ↵ aborted);
    }
    pthread_exit(0);
    return 0;
}
65
struct channel *channel_start(const char *name) {
    struct channel *channel = calloc(1, sizeof(struct channel));
    // clear YUV to black
    memset(&channel->image[0], 0, VIDEO_WIDTH * VIDEO_HEIGHT);
70     memset(&channel->image[VIDEO_WIDTH * VIDEO_HEIGHT], 128, 2 * VIDEO_WIDTH/2 *
        ↵ VIDEO_HEIGHT/2);
    channel->name = strdup(name);
    channel->width = VIDEO_WIDTH;
    channel->height = VIDEO_HEIGHT;
    if (pthread_create(&channel->thread, 0, channel_main, channel)) {
75     free(channel->name);
        free(channel);
        return 0;
    }
    return channel;
80 }

int channel_stop(struct channel *channel) {
    channel->quit = 1;
    pthread_join(channel->thread, 0);
85     int r = channel->aborted;
        free(channel->name);
        free(channel);
        return r;
}

```

12 src/channel.h

```

/*
fractal-channel-hopping -- infinite fractal television zoom
Copyright (C) 2011 Claude Heiland-Allen

5   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.

10  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.

15  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 CHANNEL_H
20 #define CHANNEL_H 1

#include <pthread.h>

#include "config.h"
25
struct channel {
    pthread_t thread;
    char *name;
    int quit;
30    int aborted;
    int width;
    int height;
    unsigned char image[VIDEO_WIDTH * VIDEO_HEIGHT + 2 * VIDEO_WIDTH/2 * 2
        ↵ VIDEO_HEIGHT/2];
};
35

struct channel *channel_start(const char *name);
int channel_stop(struct channel *channel);

#endif

```

13 src/config.h

```

/*
    fractal-channel-hopping -- infinite fractal television zoom
    Copyright (C) 2011 Claude Heiland-Allen

5    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.

10   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.

15   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 CONFIG_H
20 #define CONFIG_H 1

#define VIDEO_WIDTH 256
#define VIDEO_HEIGHT 256
#define GRID_WIDTH 16
25 #define GRID_HEIGHT 16

// #define TEXTURE_SIZE 256
#define TEXTURE_X 1.0 f
#define TEXTURE_Y 0.5625 f

```

```

30  #define CHANNELCOUNT_MAX 16

    #define OUTPUT_WIDTH 1024
    #define OUTPUT_HEIGHT 576
35  #endif

```

14 src/fch.frag

```

/*
    fractal-channel-hopping -- infinite fractal television zoom
    Copyright (C) 2011,2019 Claude Heiland-Allen

5   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.

10  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.

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

#version 130

20  uniform sampler2D images; // 4x4 tile sheet
    uniform vec3 matches[12]; // hardcoded maximum channels
    uniform int focus;
    uniform float blend0;
25  uniform float blend1;
    uniform float blend2;

    vec3 get(int n, vec2 p, float bias, bool wrap) {
        int j = int(floor(float(n) / 4.0));
30    int i = n - 4 * j;
        vec2 p1 = p;
        if (wrap) {
            float k = pow(2.0, 8.0 - bias);
            p1 = floor(p1 * k) / k;
35    }
        p1 *= 254.0 / 256.0;
        p1 += 1.0 / 256.0;
        vec2 q = (vec2(float(i), float(j)) + p1) / 4.0;
        return textureLod(images, q, bias).rgb;
40 }

    int match(int def, vec3 c) {
        float d = 1024.0;
        int m = def;
45    for (int i = 0; i < 12; ++i) {
        if (length(matches[i]) > 0.0) {
            float d2 = length(c - matches[i]);

```

```

        if (d2 < d) {
            d = d2;
50         m = i;
        }
    }
}
return m;
55 }

void main(void) {
    int f0 = focus;
    vec2 p0 = gl_TexCoord[0].xy;
60    vec3 c0 = get(f0, p0, 0.0, false);
    vec3 ca = get(f0, p0, 4.0, true);

    int f1 = match(f0, ca);
    vec2 p1 = 16.0 * p0; p1 -= floor(p1);
65    vec3 c1 = get(f1, p1, 0.0, false);
    vec3 cb = get(f1, p1, 4.0, true);

    int f2 = match(f1, cb);
    vec2 p2 = 16.0 * p1; p2 -= floor(p2);
70    vec3 c2 = get(f2, p2, 0.0, false);

    float channel[12];
    channel[0] = 0.0;
    channel[1] = 0.0;
75    channel[2] = 0.0;
    channel[3] = 0.0;
    channel[4] = 0.0;
    channel[5] = 0.0;
    channel[6] = 0.0;
80    channel[7] = 0.0;
    channel[8] = 0.0;
    channel[9] = 0.0;
    channel[10] = 0.0;
    channel[11] = 0.0;
85    channel[f0] += blend0;
    channel[f1] += blend1;
    channel[f2] += blend2;
    gl_FragData[0] = vec4(blend0 * c0 + blend1 * c1 + blend2 * c2, 1.0);
    gl_FragData[1] = vec4(channel[0], channel[1], channel[2], channel[3]);
90    gl_FragData[2] = vec4(channel[4], channel[5], channel[6], channel[7]);
    gl_FragData[3] = vec4(channel[8], channel[9], channel[10], channel[11]);
}

```

15 src/list.c

```

/*
fractal-channel-hopping -- infinite fractal television zoom
Copyright (C) 2011 Claude Heiland-Allen

5  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.

```

```
10    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 .

15    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 <assert.h>
20  #include "list.h"

void list_init(struct list *l) {
    assert(l);
    l->head = &(l->headNode);
25  l->tail = &(l->tailNode);
    l->head->pred = 0;
    l->head->next = l->tail;
    l->tail->pred = l->head;
    l->tail->next = 0;
30 }

int list_ishead(struct node *n) {
    assert(n);
    return !n->pred;
35 }

int list_istail(struct node *n) {
    assert(n);
    return !n->next;
40 }

int list_isempty(struct list *l) {
    assert(l && l->head && l->tail);
    return l->head->next == l->tail;
45 }

int list_length(struct list *l) {
    struct node *n = l->head->next;
    int i = 0;
50  while (n != l->tail) {
        n = n->next;
        i++;
    }
    return i;
55 }

void list_remove(struct node *n) {
    assert(n && n->pred && n->next);
    n->pred->next = n->next;
60  n->next->pred = n->pred;
    n->next = 0;
    n->pred = 0;
}

65 void list_insertbefore(struct node *n, struct node *beforethis) {
    assert(n && beforethis && beforethis->pred);
```



```

    n->next = beforethis;
    n->pred = beforethis->pred;
    beforethis->pred->next = n;
70   beforethis->pred = n;
    }

void list_insertafter(struct node *n, struct node *afterthis) {
    assert(n && afterthis && afterthis->next);
75   n->next = afterthis->next;
    n->pred = afterthis;
    afterthis->next->pred = n;
    afterthis->next = n;
    }
80

void list_inserttail(struct list *l, struct node *n) {
    assert(l);
    list_insertbefore(n, l->tail);
    }
85

struct node *list_removehead(struct list *l) {
    assert(l && ! list_isempty(l));
    struct node *n = l->head->next;
    list_remove(n);
90   return n;
    }

```

16 src/list.h

```

/*
   fractal-channel-hopping -- infinite fractal television zoom
   Copyright (C) 2011 Claude Heiland-Allen

5   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.

10  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.

15  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 LIST_H
20  #define LIST_H 1

    struct node {
        struct node *next;
        struct node *pred;
25  };

    struct list {
        struct node *head;
        struct node *tail;

```

```

30     // private
        struct node headNode;
        struct node tailNode;
    };

35     void list_init(struct list *l);
    int list_ishead(struct node *n);
    int list_istail(struct node *n);
    int list_isempty(struct list *l);
    int list_length(struct list *l);
40     void list_remove(struct node *n);
    void list_insertbefore(struct node *n, struct node *beforethis);
    void list_insertafter(struct node *n, struct node *afterthis);
    void list_inserttail(struct list *l, struct node *n);
    struct node *list_removehead(struct list *l);
45
#endif

```

17 src/main.c

```

/*
    fractal-channel-hopping -- infinite fractal television zoom
    Copyright (C) 2011,2019 Claude Heiland-Allen

5     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.

10    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.

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

#define _DEFAULT_SOURCE
20 #define _POSIX_C_SOURCE 200809

#include <assert.h>
#include <math.h>
#include <signal.h>
25 #include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>

30 #include <GL/glew.h>
#include <GL/glut.h>

#include <jack/jack.h>
35
#include "channel.h"
#include "record.h"

```

```

#include "fch.frag.c"
#include "yuv2rgb.frag.c"
40
int max(int x, int y) {
    return x > y ? x : y;
}

45 unsigned int roundtwo(unsigned int x) {
    assert(x <= 1u << 31u); // termination condition
    unsigned int y = 1;
    while (y < x) y <<= 1;
    return y;
50 }

unsigned int logtwo(unsigned int x) {
    assert(x <= 1u << 31u); // termination condition
    unsigned int y = 1, z = 0;
55 while (y < x) { y <<= 1; z += 1; };
    return z;
}

int focus = 0;
60 int speed = 75;
int frames = 0;
int tframes = 0;

int count;
65 struct channel **channels;

struct record *recorder = 0;

int winw, winh;
70 int tsize;
GLuint texiny, texinu, texinv, timages, toutput, tmatch[3]; // hardcoded: (12) ↗
    ↘ channels / 4
GLuint fbo;

// fractalization shader
75 GLhandleARB prog;
GLhandleARB frag;
GLint uimages, umatches, ufocus, ublend0, ublend1, ublend2;
GLfloat vmatches[4 * 4][3];

80 // yuv2rgb shader
GLhandleARB prog_yuv2rgb;
GLhandleARB frag_yuv2rgb;
GLint yuv2rgb_y, yuv2rgb_u, yuv2rgb_v;

85 double zoom;
int zoomi;
int zoomj;
double blend0;
double blend1;
90 double blend2;

jack_client_t *jclient;
jack_port_t *jporto[2];

```

```

jack_port_t *jport[CHANNEL_COUNT_MAX][2];
95 float jlevel[2][CHANNEL_COUNT_MAX];
int jwhich = 0;

void errorcb(const char *desc) {
    fprintf(stderr, "JACK error: %s\n", desc);
100 }

void shutdowncb(void *arg) {
    exit(1);
}

105 int processcb(jack_nframes_t nframes, void *arg) {
    jack_default_audio_sample_t *in[CHANNEL_COUNT_MAX][2], *out[2];
    for (int c = 0; c < count; ++c) {
        for (int k = 0; k < 2; ++k) {
110             in[c][k] = (jack_default_audio_sample_t *) jack_port_get_buffer(jport[c][k],
                ↪ nframes);
        }
    }
    out[0] = (jack_default_audio_sample_t *) jack_port_get_buffer(jport[0], ↪
        ↪ nframes);
    out[1] = (jack_default_audio_sample_t *) jack_port_get_buffer(jport[1], ↪
        ↪ nframes);
115 for (jack_nframes_t i = 0; i < nframes; ++i) {
    for (int k = 0; k < 2; ++k) {
        jack_default_audio_sample_t o = 0;
        for (int c = 0; c < count; ++c) {
            o += jlevel[jwhich][c] * in[c][k][i];
120        }
        out[k][i] = o;
    }
}
return 0;
125 }

void realloctexture(GLuint t, GLenum fmt, float r, float g, float b) {
    // resize texture
    glBindTexture(GL_TEXTURE_2D, t);
130 glTexImage2D(GL_TEXTURE_2D, 0, fmt, tsize, tsize, 0, fmt, GL_UNSIGNED_BYTE, 0)↪
        ↪ ;
    glBindTexture(GL_TEXTURE_2D, 0);
    // clear texture
    glViewport(0, 0, tsize, tsize);
    glBindFramebufferEXT(GL_FRAMEBUFFER_EXT, fbo);
135 glFramebufferTexture2D(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT0_EXT, ↪
        ↪ GL_TEXTURE_2D, t, 0);
    GLenum dbs[] = { GL_COLOR_ATTACHMENT0_EXT };
    glDrawBuffers(1, dbs);
    glClearColor(r, g, b, 0);
    glClear(GL_COLOR_BUFFER_BIT);
140 glFramebufferTexture2D(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT0_EXT, ↪
        ↪ GL_TEXTURE_2D, 0, 0);
    glBindFramebufferEXT(GL_FRAMEBUFFER_EXT, 0);
}

void reshapecb(int w, int h) {

```

```

145     if (recorder && (winw != w || winh != h)) {
        record_stop(recorder);
        recorder = 0;
    }
    winw = w;
150    winh = h;
    int oldtsize = tsize;
    tsize = roundtwo(max(w, h));
    if (oldtsize != tsize) {
        realloctexture(toutput, GL_RGBA, 0, 0, 0);
155        realloctexture(tmatch[0], GL_RGBA, 0, 0, 0);
        realloctexture(tmatch[1], GL_RGBA, 0, 0, 0);
        realloctexture(tmatch[2], GL_RGBA, 0, 0, 0);
    }
}

160 void displaycb(void) {
    // upload channels of video to tile sheet
    {
        // upload YUV planes
165        glActiveTexture(GL_TEXTURE0);
        glBindTexture(GL_TEXTURE_2D, texiny);
        for (int c = 0; c < count; ++c) {
            glTexSubImage2D(GL_TEXTURE_2D, 0, (c % 4) * VIDEO_WIDTH, (c / 4) * ↵
                ↵ VIDEO_HEIGHT, VIDEO_WIDTH, VIDEO_HEIGHT, GL_LUMINANCE, ↵
                ↵ GL_UNSIGNED_BYTE, &channels[c]->image[0]);
        }
170        glActiveTexture(GL_TEXTURE1);
        glBindTexture(GL_TEXTURE_2D, texinu);
        for (int c = 0; c < count; ++c) {
            glTexSubImage2D(GL_TEXTURE_2D, 0, (c % 4) * VIDEO_WIDTH/2, (c / 4) * ↵
                ↵ VIDEO_HEIGHT/2, VIDEO_WIDTH/2, VIDEO_HEIGHT/2, GL_LUMINANCE, ↵
                ↵ GL_UNSIGNED_BYTE, &channels[c]->image[VIDEO_WIDTH * VIDEO_HEIGHT]);
        }
175        glActiveTexture(GL_TEXTURE2);
        glBindTexture(GL_TEXTURE_2D, texinv);
        for (int c = 0; c < count; ++c) {
            glTexSubImage2D(GL_TEXTURE_2D, 0, (c % 4) * VIDEO_WIDTH/2, (c / 4) * ↵
                ↵ VIDEO_HEIGHT/2, VIDEO_WIDTH/2, VIDEO_HEIGHT/2, GL_LUMINANCE, ↵
                ↵ GL_UNSIGNED_BYTE, &channels[c]->image[VIDEO_WIDTH * VIDEO_HEIGHT + ↵
                ↵ VIDEO_WIDTH/2 * VIDEO_HEIGHT/2]);
        }
180        // convert to RGB
        glViewport(0, 0, 1024, 1024);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        glOrtho(0, 1, 0, 1, -1, 1);
185        glBindFramebufferEXT(GL_FRAMEBUFFER_EXT, fbo);
        glFramebufferTexture2D_EXT(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT0_EXT, ↵
            ↵ GL_TEXTURE_2D, timages, 0);
        GLenum dbs[] = { GL_COLOR_ATTACHMENT0_EXT };
        glDrawBuffers(1, dbs);
        glUseProgramObjectARB(prog_yuv2rgb);
190        glUniform1i(yuv2rgb_y, 0);
        glUniform1i(yuv2rgb_u, 1);
        glUniform1i(yuv2rgb_v, 2);
        glBegin(GL_QUADS); {

```

```

195     glTexCoord2f(0, 0); glVertex2f(0, 0);
    glTexCoord2f(1, 0); glVertex2f(1, 0);
    glTexCoord2f(1, 1); glVertex2f(1, 1);
    glTexCoord2f(0, 1); glVertex2f(0, 1);
} glEnd();
glUseProgramObjectARB(0);
200 glFramebufferTexture2DEXT(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT0_EXT, ↵
    ↵ GL_TEXTURE_2D, 0, 0);
glBindFramebufferEXT(GL_FRAMEBUFFER_EXT, 0);
glBindTexture(GL_TEXTURE_2D, 0);
glActiveTexture(GL_TEXTURE1);
glBindTexture(GL_TEXTURE_2D, 0);
205 glActiveTexture(GL_TEXTURE0);
glBindTexture(GL_TEXTURE_2D, timages);
glGenerateMipmap(GL_TEXTURE_2D);
glGetTexImage(GL_TEXTURE_2D, 8 /* hardcoded log2(4*256) - 2 */, GL_RGB, ↵
    ↵ GL_FLOAT, vmatches);
glBindTexture(GL_TEXTURE_2D, 0);
210 }

// set up zooming view
glViewport(0, 0, winw, winh);
{
215     glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0, GRID_WIDTH, 0, GRID_HEIGHT, -1, 1);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
220     double dx = zoomi * GRID_WIDTH / (GRID_WIDTH - 1.0);
    double dy = GRID_HEIGHT - zoomj * GRID_HEIGHT / (GRID_HEIGHT - 1.0);
    glTranslatef(dx, dy, 0);
    glScalef(zoom, zoom, zoom);
    glTranslatef(-dx, -dy, 0);
225 }

// mega shader action
glBindFramebufferEXT(GL_FRAMEBUFFER_EXT, fbo);
glFramebufferTexture2DEXT(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT0_EXT, ↵
    ↵ GL_TEXTURE_2D, toutput, 0);
230 glFramebufferTexture2DEXT(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT1_EXT, ↵
    ↵ GL_TEXTURE_2D, tmatch[0], 0);
glFramebufferTexture2DEXT(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT2_EXT, ↵
    ↵ GL_TEXTURE_2D, tmatch[1], 0);
glFramebufferTexture2DEXT(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT3_EXT, ↵
    ↵ GL_TEXTURE_2D, tmatch[2], 0);
GLenum dbs[] = { GL_COLOR_ATTACHMENT0_EXT, GL_COLOR_ATTACHMENT1_EXT, ↵
    ↵ GL_COLOR_ATTACHMENT2_EXT, GL_COLOR_ATTACHMENT3_EXT };
glDrawBuffers(4, dbs);
235 glUseProgramObjectARB(prog);
glBindTexture(GL_TEXTURE_2D, timages);
glUniform1i(uimages, 0);
glUniform3fv(umatches, 12, &vmatches[0][0]);
glUniform1i(ufocus, focus);
240 glUniform1f(ublend0, blend0);
glUniform1f(ublend1, blend1);
glUniform1f(ublend2, blend2);
glBegin(GL_QUADS); {

```

```

    glTexCoord2f(0, 1); glVertex2f(0, 0);
245    glTexCoord2f(1, 1); glVertex2f(GRID_WIDTH, 0);
    glTexCoord2f(1, 0); glVertex2f(GRID_WIDTH, GRID_HEIGHT);
    glTexCoord2f(0, 0); glVertex2f(0, GRID_HEIGHT);
} glEnd();
glBindTexture(GL_TEXTURE_2D, 0);
250 glUseProgramObjectARB(0);
glFramebufferTexture2D(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT0_EXT, ↵
    ↵ GL_TEXTURE_2D, 0, 0);
glFramebufferTexture2D(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT1_EXT, ↵
    ↵ GL_TEXTURE_2D, 0, 0);
glFramebufferTexture2D(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT2_EXT, ↵
    ↵ GL_TEXTURE_2D, 0, 0);
glFramebufferTexture2D(GL_FRAMEBUFFER_EXT, GL_COLOR_ATTACHMENT3_EXT, ↵
    ↵ GL_TEXTURE_2D, 0, 0);
255 glBindFramebufferEXT(GL_FRAMEBUFFER_EXT, 0);

// show output image
glLoadIdentity();
glBindTexture(GL_TEXTURE_2D, toutput);
260 glBegin(GL_QUADS); {
    float tx = winw * 1.0f / tsize;
    float ty = winh * 1.0f / tsize;
    glTexCoord2f(0, 0); glVertex2f(0, 0);
    glTexCoord2f(tx, 0); glVertex2f(GRID_WIDTH, 0);
265    glTexCoord2f(tx, ty); glVertex2f(GRID_WIDTH, GRID_HEIGHT);
    glTexCoord2f(0, ty); glVertex2f(0, GRID_HEIGHT);
} glEnd();
glBindTexture(GL_TEXTURE_2D, 0);

270 // grab auxiliary images to audio mixer
glBindTexture(GL_TEXTURE_2D, tmatch[0]);
glGenerateMipmap(GL_TEXTURE_2D);
glGetTexImage(GL_TEXTURE_2D, logtwo(tsize), GL_RGBA, GL_FLOAT, &jlevel[1 - ↵
    ↵ jwhich][0]);
glBindTexture(GL_TEXTURE_2D, tmatch[1]);
275 glGenerateMipmap(GL_TEXTURE_2D);
glGetTexImage(GL_TEXTURE_2D, logtwo(tsize), GL_RGBA, GL_FLOAT, &jlevel[1 - ↵
    ↵ jwhich][4]);
glBindTexture(GL_TEXTURE_2D, tmatch[2]);
glGenerateMipmap(GL_TEXTURE_2D);
glGetTexImage(GL_TEXTURE_2D, logtwo(tsize), GL_RGBA, GL_FLOAT, &jlevel[1 - ↵
    ↵ jwhich][8]);
280 glBindTexture(GL_TEXTURE_2D, 0);
double s = 0;
for (int c = 0; c < count; ++c) {
    s += jlevel[1 - jwhich][c];
}
285 if (!s) {
    s = 1;
}
for (int c = 0; c < count; ++c) {
    jlevel[1 - jwhich][c] /= s;
290 }
jwhich = 1 - jwhich;

// PPM recording on stdout

```

```

    if (recorder) {
295         record_frame(recorder);
    }

    glutSwapBuffers();

300    // bug free code?
    glutReportErrors();

    // we done a frame
    tframes++;
305    frames++;
}

float smatch[4][16][4][16][3];
float matches[4][4][3];

310    struct timespec clock0;
    struct timespec clock1;

void timercb(int v) {
315    glutTimerFunc(1, timercb, v);
    clock_gettime(CLOCK_REALTIME, &clock1);
    double dt = (clock1.tv_sec - clock0.tv_sec) + 1.0e-9 * (clock1.tv_nsec -
        ↵ clock0.tv_nsec);
    if (dt < 0.04 * tframes) {
        return;
320    }
    if (frames == speed) {
        // refocus
        glBindTexture(GL_TEXTURE_2D, timages);
        glGenerateMipmap(GL_TEXTURE_2D);
325        glGetTexImage(GL_TEXTURE_2D, 4, GL_RGB, GL_FLOAT, &smatch[0][0][0][0][0]);
        glGetTexImage(GL_TEXTURE_2D, 8, GL_RGB, GL_FLOAT, &matches[0][0][0]);
        glBindTexture(GL_TEXTURE_2D, 0);
        float ml = 65536.0;
        int mi = focus;
330        for (int c = 0; c < count; ++c) {
            float s = 0;
            for (int k = 0; k < 3; ++k) {
                float ds = smatch[focus / 4][zoomj][focus % 4][zoomi][k] - vmatches[c][k]
                    ↵ ];
                s += ds * ds;
335            }
            if (s < ml) {
                ml = s;
                mi = c;
            }
340        }
        focus = mi;
        ml = 65536.0;
        for (int c = 0; c < count; ++c) {
            float s = 0;
345            for (int k = 0; k < 3; ++k) {
                float ds = matches[focus / 4][focus % 4][k] - vmatches[c][k];
                s += ds * ds;
            }
        }
    }
}

```



```

        if (s < ml) {
350             ml = s;
                mi = c;
        }
    }
    focus = mi;
355    // self-centering random walk
    if (rand() % 8) {
        int coin = rand() % (GRID_WIDTH - 1);
        zoomi += coin < zoomi ? -1 : 1;
    }
360    if (rand() % 8) {
        int coin = rand() % (GRID_HEIGHT - 1);
        zoomj += coin < zoomj ? -1 : 1;
    }
    frames = 0;
365 }

// zoom blending
double k = frames * 1.0 / speed;
zoom = pow(16, k); // hardcoded power - grid size ...
370 blend2 = 1 - cos(2 * 3.141592653 * (k + 0) / 3);
    blend1 = 1 - cos(2 * 3.141592653 * (k + 1) / 3);
    blend0 = 1 - cos(2 * 3.141592653 * (k + 2) / 3);
    double blendt = blend0 + blend1 + blend2;
    blend0 /= blendt;
375    blend1 /= blendt;
    blend2 /= blendt;

    glutPostRedisplay();
}
380
int fullscreen = 0;

void keyboardcb(unsigned char key, int x, int y) {
    switch (key) {
385        case 'R':
            if (recorder) {
                record_stop(recorder);
                recorder = 0;
            } else {
390                recorder = record_start(winw, winh, jclient, jporto);
            }
            break;
        case 'Q':
        case 27:
395            exit(0);
            break;
    }
}

400 void keyspecialcb(int key, int x, int y) {
    switch (key) {
        case GLUT_KEY_F11:
            fullscreen = !fullscreen;
            if (fullscreen) {
405                glutFullScreen();
            }
    }
}

```

```

        glutSetCursor(GLUT_CURSOR_NONE);
    } else {
        glutReshapeWindow(OUTPUT_WIDTH, OUTPUT_HEIGHT);
        glutSetCursor(GLUT_CURSOR_INHERIT);
410    }
        break;
    }
}

415 void exitcb(void) {
    if (recorder) {
        record_stop(recorder);
        recorder = 0;
    }
420    jack_client_close(jclient);
    killpg(getpgrp(), SIGKILL); // kill all our processes
    for (int c = 0; c < count; ++c) {
        channel_stop(channels[c]);
    }
425    free(channels);
}

int main(int argc, char **argv) {

430    // initialisation
    if (argc <= 1) { return 1; }
    count = argc - 1;
    channels = calloc(count, sizeof(struct channel *));

435    srand(time(0));
    zoomj = rand() % GRID_HEIGHT;
    zoomi = rand() % GRID_WIDTH;
    glutInitWindowSize(OUTPUT_WIDTH, OUTPUT_HEIGHT);
    glutInit(&argc, argv);
440    glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE);
    glutCreateWindow("fractal-channel-hopping");
    glewInit();

    // set up jack first

445    jack_set_error_function(errorcb);
    if (!(jclient = jack_client_open("fch", 0, 0))) {
        fprintf(stderr, "jack server not running?\n");
        return 1;
450    }
    jack_set_process_callback(jclient, processcb, 0);
    jack_on_shutdown(jclient, shutdowncb, 0);
    for (int c = 0; c < count; ++c) {
        for (int k = 0; k < 2; ++k) {
455            char namebuf[64];
            snprintf(namebuf, 62, "%s_%d", argv[c + 1], k + 1);
            namebuf[63] = 0;
            jport[c][k] = jack_port_register(jclient, namebuf, JACK_DEFAULT_AUDIO_TYPE,
                ↵, JackPortIsInput, 0);
        }
460    }
    for (int k = 0; k < 2; ++k) {

```

```

    char namebuf[64];
    snprintf(namebuf, 62, "output_%d", k + 1);
    namebuf[63] = 0;
465     jporto[k] = jack_port_register(jclient, namebuf, JACK_DEFAULT_AUDIO_TYPE,
        ↵ JackPortIsOutput, 0);
    }
    if (jack_activate(jclient)) {
        fprintf(stderr, "cannot activate JACK client");
        return 1;
470     }
    jack_connect(jclient, "fch:output_1", "system:playback_1");
    jack_connect(jclient, "fch:output_2", "system:playback_2");

    // then start streaming channels which connect to us via jack
475     for (int c = 0; c < count; ++c) {
        channels[c] = channel_start(argv[c + 1]);
    }

480     // and prepare the display

    glHint(GL_GENERATE_MIPMAP_HINT, GL_FASTEST);
    glEnable(GL_TEXTURE_2D);
    glGenFramebuffersEXT(1, &fbo);
485     // YUV planar video
    tsize = 1024;
    glGenTextures(1, &texiny);
    realloctexture(texiny, GL_LUMINANCE, 0, 0, 0);
490     glBindTexture(GL_TEXTURE_2D, texiny);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
    glBindTexture(GL_TEXTURE_2D, 0);
    tsize /= 2;
495     glGenTextures(1, &texinu);
    realloctexture(texinu, GL_LUMINANCE, 0.5, 0.5, 0.5);
    glBindTexture(GL_TEXTURE_2D, texinu);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
500     glBindTexture(GL_TEXTURE_2D, 0);
    glGenTextures(1, &texinv);
    realloctexture(texinv, GL_LUMINANCE, 0.5, 0.5, 0.5);
    glBindTexture(GL_TEXTURE_2D, texinv);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
505     glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
    glBindTexture(GL_TEXTURE_2D, 0);

    // RGB interleaved video
    tsize = 1024;
510     glGenTextures(1, &timages);
    realloctexture(timages, GL_RGBA, 0, 0, 0);
    glBindTexture(GL_TEXTURE_2D, timages);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR_MIPMAP_LINEAR) ↵
        ↵ ;
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
515     glBindTexture(GL_TEXTURE_2D, 0);

```

```

// RGB output
tsize = roundtwo(max(OUTPUT_WIDTH, OUTPUT_HEIGHT));
glGenTextures(1, &toutput);
520 realloctexture(toutput, GL_RGBA, 0, 0, 0);
glBindTexture(GL_TEXTURE_2D, toutput);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glBindTexture(GL_TEXTURE_2D, 0);
525 glGenTextures(3, &tmatch[0]);
realloctexture(tmatch[0], GL_RGBA, 0, 0, 0);
realloctexture(tmatch[1], GL_RGBA, 0, 0, 0);
realloctexture(tmatch[2], GL_RGBA, 0, 0, 0);

530 // fractalization shader
GLint success;
prog = glCreateProgramObjectARB();
frag = glCreateShaderObjectARB(GL_FRAGMENT_SHADER_ARB);
glShaderSourceARB(frag, 1, &fch_frag, 0);
535 glCompileShaderARB(frag);
glAttachObjectARB(prog, frag);
glLinkProgramARB(prog);
glGetObjectParameterivARB(prog, GL_OBJECT_LINK_STATUS_ARB, &success);
if (!success) {
540     GLhandleARB obj = prog;
     int infologLength = 0;
     int maxLength;
     if (glIsShader(obj)) {
         glGetShaderiv(obj, GL_INFO_LOG_LENGTH, &maxLength);
545     } else {
         glGetProgramiv(obj, GL_INFO_LOG_LENGTH, &maxLength);
     }
     char *infoLog = malloc(maxLength);
     if (!infoLog) {
550         exit(1);
     }
     if (glIsShader(obj)) {
         glGetShaderInfoLog(obj, maxLength, &infologLength, infoLog);
     } else {
555         glGetProgramInfoLog(obj, maxLength, &infologLength, infoLog);
     }
     if (infologLength > 0) {
         fprintf(stderr, "%s\n", infoLog);
     }
560     free(infoLog);
     exit(1);
}
uimages = glGetUniformLocationARB(prog, "images");
umatches = glGetUniformLocationARB(prog, "matches");
565 ufocus = glGetUniformLocationARB(prog, "focus");
ublend0 = glGetUniformLocationARB(prog, "blend0");
ublend1 = glGetUniformLocationARB(prog, "blend1");
ublend2 = glGetUniformLocationARB(prog, "blend2");

570 // YUV2RGB shader
prog_yuv2rgb = glCreateProgramObjectARB();
frag_yuv2rgb = glCreateShaderObjectARB(GL_FRAGMENT_SHADER_ARB);
glShaderSourceARB(frag_yuv2rgb, 1, &yuv2rgb_frag, 0);

```

```

575     glCompileShaderARB(frag_yuv2rgb);
    glAttachObjectARB(prog_yuv2rgb, frag_yuv2rgb);
    glLinkProgramARB(prog_yuv2rgb);
    glGetObjectParameterivARB(prog_yuv2rgb, GL_OBJECT_LINK_STATUS_ARB, &success);
    if (! success) {
        GLhandleARB obj = prog_yuv2rgb;
580     int infologLength = 0;
        int maxLength;
        if (glIsShader(obj)) {
            glGetShaderiv(obj, GL_INFO_LOG_LENGTH, &maxLength);
        } else {
585     glGetProgramiv(obj, GL_INFO_LOG_LENGTH, &maxLength);
        }
        char *infoLog = malloc(maxLength);
        if (!infoLog) {
            exit(1);
590     }
        if (glIsShader(obj)) {
            glGetShaderInfoLog(obj, maxLength, &infologLength, infoLog);
        } else {
            glGetProgramInfoLog(obj, maxLength, &infologLength, infoLog);
595     }
        if (infologLength > 0) {
            fprintf(stderr, "%s\n", infoLog);
        }
        free(infoLog);
600     exit(1);
    }
    yuv2rgb_y = glGetUniformLocationARB(prog_yuv2rgb, "y");
    yuv2rgb_u = glGetUniformLocationARB(prog_yuv2rgb, "u");
    yuv2rgb_v = glGetUniformLocationARB(prog_yuv2rgb, "v");
605
    // callbacks
    glutKeyboardFunc(keyboardcb);
    glutSpecialFunc(keyspecialcb);
    glutReshapeFunc(reshapecb);
610    glutDisplayFunc(displaycb);
    glutTimerFunc(1, timercb, 1);
    atexit(exitcb);

    // main loop
615    clock_gettime(CLOCK_REALTIME, &clock0);
    glutMainLoop();
    return 0;
}

```

18 src/Makefile

```

# fractal-channel-hopping -- infinite fractal television zoom
# Copyright (C) 2011 Claude Heiland-Allen
#
# This program is free software: you can redistribute it and/or modify
5 # 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,

```

```

10  # 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
15  # along with this program. If not, see <http://www.gnu.org/licenses/>.

CC=gcc
CFLAGS=-std=c99 -Wall -pedantic -Wextra -Wno-unused-parameter -O3 -march=native ↵
    ↵ -pthread -lGLEW -lGL -lglut -ljack -lm

20  BINARY = fractal-channel-hopping
    SOURCES = channel.c record.c pfifo.c list.c main.c
    HEADERS = channel.h record.h pfifo.h list.h config.h
    GENHEAD = fch.frag.c yuv2rgb.frag.c

25  all: $(BINARY)

    clean:
        -rm $(BINARY) $(GENHEAD)

30  .SUFFIXES:
    .PHONY: all clean

$(BINARY): $(SOURCES) $(HEADERS) $(GENHEAD)
        $(CC) $(CFLAGS) -o $(BINARY) $(SOURCES)

35  %.frag.c: %.frag s2c.sh
        ./s2c.sh $_frag < $< > $@

```

19 src/pfifo.c

```

/*
    fractal-channel-hopping -- infinite fractal television zoom
    Copyright (C) 2011 Claude Heiland-Allen

5   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.

10  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.

15  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 <assert.h>
20  #include <string.h>
    #include "pfifo.h"

    void *pfifo_consumerthread(void *);

25  struct pfifo *pfifo_create(pfifo_consumer *consumer, void *consumerdata) {

```

```

    struct pfifo *p = malloc(sizeof(struct pfifo));
    if (!p) return 0;
    list_init(&(p->list));
    p->running = 1;
30  p->consumer = consumer;
    p->consumerdata = consumerdata;
    p->mutex = malloc(sizeof(pthread_mutex_t));
    pthread_mutex_init(p->mutex, 0);
    p->nonempty = malloc(sizeof(pthread_cond_t));
35  pthread_cond_init(p->nonempty, 0);
    pthread_create(&(p->thread), 0, pfifo_consumerthread, p);
    return p;
}

40  void pfifo_destroy(struct pfifo *p) {
    p->running = 0;
    pthread_join(p->thread, 0);
    // FIXME proper cleanup...
}

45  void pfifo_enqueue(struct pfifo *p, size_t length, const void *data) {
    struct pfifo_node *n = malloc(sizeof(struct pfifo_node));
    assert(n);
    n->length = length;
50  n->data = malloc(length);
    assert(n->data);
    memcpy(n->data, data, length);
    pthread_mutex_lock(p->mutex);
    list_inserttail(&(p->list), &(n->node));
55  pthread_mutex_unlock(p->mutex);
    pthread_cond_signal(p->nonempty);
}

void *pfifo_consumerthread(void *fifo) {
60  struct pfifo *p = fifo;
    pthread_mutex_lock(p->mutex);
    while (p->running) {
        while (list_isempty(&(p->list))) pthread_cond_wait(p->nonempty, p->mutex);
        struct pfifo_node *n = (struct pfifo_node *) list_removehead(&(p->list));
65  p->consumer(p->consumerdata, n->length, n->data);
        free(n->data);
        free(n);
    }
    pthread_exit(0);
70  return 0;
}

```

20 src/pfifo.h

```

/*
fractal-channel-hopping -- infinite fractal television zoom
Copyright (C) 2011 Claude Heiland-Allen

```

```

5  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.

```

```

10   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.

15   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 PFIFO_H
20   #define PFIFO_H 1

    #include <stdlib.h>
    #include <pthread.h>
    #include "list.h"

25   typedef void (pfifo_consumer)(void *, size_t, void *);

    struct pfifo_node {
        struct node node;
30     size_t length;
        void *data;
    };

    struct pfifo {
35     pfifo_consumer *consumer;
        void *consumerdata;
        struct list list;
        pthread_t thread;
        pthread_mutex_t *mutex;
40     pthread_cond_t *nonempty;
        int running;
    };

    struct pfifo *pfifo_create(pfifo_consumer *consumer, void *consumerdata);
45 void pfifo_destroy(struct pfifo *fifo);
    void pfifo_enqueue(struct pfifo *fifo, size_t length, const void *data);

    #endif

```

21 src/record.c

```

/*
    fractal-channel-hopping -- infinite fractal television zoom
    Copyright (C) 2011,2019 Claude Heiland-Allen

5   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.

10  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.

```



```

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

#define _DEFAULT_SOURCE

20     #include <stdio.h>
    #include <stdlib.h>
    #include <string.h>
    #include <time.h>

25     #include <GL/glew.h>
    #include <jack/jack.h>

    #include "pfifo.h"
30     #include "record.h"

    void record_writer(void *arg, size_t l, void *data) {
        struct record *record = arg;
        unsigned char *buffer = data;
35         fwrite(record->header, strlen(record->header), 1, record->ppm);
        for (int y = record->height - 1; y >= 0; --y) {
            fwrite(buffer + record->width * y * 3, record->width * 3, 1, record->ppm);
        }
        if (! jack_port_connected_to(record->jport[0], "record:input_1")) {
40             jack_connect(record->jclient, jack_port_name(record->jport[0]), "record:↵
                ↵ input_1");
        }
        if (! jack_port_connected_to(record->jport[1], "record:input_2")) {
            jack_connect(record->jclient, jack_port_name(record->jport[1]), "record:↵
                ↵ input_2");
        }
45     }

    struct record *record_start(int w, int h, jack_client_t *jclient, jack_port_t **↵
        ↵ jports) {
        struct record *record = malloc(sizeof(struct record));
        record->width = w;
50         record->height = h;
        record->bytes = w * h * 3;
        record->buffer = malloc(record->bytes);
        snprintf(record->header, 62, "P6↵n%d %d 255↵n", w, h);
        record->header[63] = 0;
55         const char *vfmt = "ppmtov4m -v0 -S444 -F25:1 2>'tmp/record.ppm.log' | ffmpeg ↵
            ↵ -loglevel 0 -f yuv4mpegpipe -i pipe:- -ac 2 -f jack -i record -target ↵
            ↵ pal-dvd -shortest 'fractal-channel-hopping-%Y-%m-%d.%H-%M-%S-%z.mpeg' >'↵
            ↵ tmp/record.ffmpeg.log' 2>&1";
        int vlen = strlen(vfmt) + 64;
        char *vcmd = malloc(vlen);
        time_t t = time(NULL);
        struct tm tm;
60         localtime_r(&t, &tm);
        if (0 == strftime(vcmd, vlen, vfmt, &tm)) {
            return 0;
        }
        record->jclient = jclient;
65         record->jport[0] = jports[0];

```

```

    record->jport[1] = jports[1];
    record->ppm = popen(vcmd, "w");
    record->pfifo = pfifo_create(record_writer, record);
    return record;
70 }

void record_frame(struct record *record) {
    glReadPixels(0, 0, record->width, record->height, GL_RGB, GL_UNSIGNED_BYTE, ↵
        record->buffer);
    pfifo_enqueue(record->pfifo, record->bytes, record->buffer);
75 }

void record_stop(struct record *record) {
    pfifo_destroy(record->pfifo);
    pclose(record->ppm);
80 free(record->buffer);
    free(record);
}

```

22 src/record.h

```

/*
    fractal-channel-hopping -- infinite fractal television zoom
    Copyright (C) 2011 Claude Heiland-Allen

5   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.

10  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.

15  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 RECORD_H
20 #define RECORD_H 1

#include <stdio.h>

#include <jack/jack.h>
25 #include "pfifo.h"

struct record {
    int width;
30 int height;
    int bytes;
    unsigned char *buffer;
    char header[64];
    FILE *ppm;
35 FILE *wav;
    struct pfifo *pfifo;
}

```

```

    jack_client_t *jclient;
    jack_port_t *jport[2];
};
40 struct record *record_start(int w, int h, jack_client_t *jclient, jack_port_t **
    ↵ jports);
void record_frame(struct record *record);
void record_stop(struct record *record);
45 #endif

```

23 src/s2c.sh

```

#!/bin/bash
# fractal-channel-hopping -- infinite fractal television zoom
# Copyright (C) 2011 Claude Heiland-Allen
#
5 # 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.
#
10 # 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.
#
15 # You should have received a copy of the GNU General Public License
# along with this program. If not, see <http://www.gnu.org/licenses/>.
echo "/* machine-generated file, do not edit */"
echo "static const char *$1 ="
sed 's|^|"'|'|
20 sed 's|$|\\n"'|'|
echo ";"

```

24 src/yuv2rgb.frag

```

/*
    fractal-channel-hopping -- infinite fractal television zoom
    Copyright (C) 2011 Claude Heiland-Allen
5    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.
10    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.
15    You should have received a copy of the GNU General Public License
    along with this program. If not, see <http://www.gnu.org/licenses/>.
*/
#version 130

```

```

20  uniform sampler2D y;
    uniform sampler2D u;
    uniform sampler2D v;

25  void main(void) {
        const mat4 m = mat4(
            1.1643828125,  0.0,          1.59602734375, -0.87078515625,
            1.1643828125, -0.39176171875, -0.81296875,  0.52959375,
            1.1643828125,  2.017234375,  0.0,          -1.081390625,
30      0.0,              0.0,          0.0,          1.0
        );
        vec2 p = gl_TexCoord[0].xy;
        vec3 yuv = vec3(texture(y, p).r, texture(u, p).r, texture(v, p).r);
        gl_FragData[0] = vec4(yuv, 1.0) * m;
35  }

```

25 start.sh

```

#!/bin/bash
# fractal-channel-hopping -- infinite fractal television zoom
# Copyright (C) 2011,2019 Claude Heiland-Allen
#
5  # 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.
#
10 # 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.
#
15 # You should have received a copy of the GNU General Public License
#   along with this program. If not, see <http://www.gnu.org/licenses/>.
#
S="$(dirname "$(readlink -f "$0")")"
FCH="$S/src/fractal-channel-hopping"
20 CHANNELS=(Channel*)
make -C "$S/src" &&
mkdir -p tmp &&
for c in "${CHANNELS[@]}"
do
25   rm -f "tmp/${c}.fifo"
   mkfifo "tmp/${c}.fifo"
done &&
"${FCH}" "${CHANNELS[@]}"

```