

Class 38: Representing Color Palettes with Vectors

Held: Wednesday, 9 April 2008

Summary: We consider *vectors*, an alternative to files and lists for storing collections of data. We also consider how vectors can be used to represent color palettes.

Related Pages:

- EBoard.
- Lab: Vectors.
- Reading: Vectors.
- Due: Assignment 8: Run-Length Encoding.

Notes:

- Normal start-of-class time for discussion of campus issues.
- EC for attending tonight's Con Brio concert (9:30 in the pub).
- EC for attending Thursday noon applied studio recital in Sebring-Lewis.
- EC for attending Thursday 4:15 talk on Calculus for Dogs.
- EC for attending Friday evening talk by Patch Adams.
- EC for attending Saturday Evening's 10:00 p.m. RHPS.
- Jonathan Tsu and Ian Young are conducting a usability study on the Grinnell CS Department website as a project for Human-Computer Interaction. We are looking for students currently taking Fundamentals of CS to take part. The study should take about twenty minutes to complete. Email [youngian] if you are interested. *EC for participating.*
- Reading for Friday: Analyzing Algorithms (ready Thursday).

Overview:

- Problems with lists.
- A solution: Vectors.
- Important vector procedures.
- Representing palettes with vectors.

List Deficiencies

- Now that we've worked with lists for a while, we've identified some things that make lists inappropriate for some situations.
 - Lists are *expensive* to use; to find the length of a list or to access an element late in the list, you need to cdr down the list.
 - Lists are *fixed*; you can't easily change an element of a list.
- At the same time, there are some advantages to lists:

- Lists are *dynamic*; it is easy to grow and shrink a list.
- Lists are *inherently recursive*; the type is defined recursively.
- Lists are *simple*; you can build almost every list operation through just a few basic operations (`car`, `cdr`, `null`, and `null?`).

An Alternative: Vectors

- Vectors provide an alternative to lists.
- They have two primary advantages:
 - Vectors are *indexed*: You can quickly access elements by number.
 - Vectors are *mutable*: You can change the elements of a vector.
- In order to obtain these benefits, vectors lack some key features of lists. In particular,
 - Vectors are *static*: Once you've created a vector, you cannot change its length.
- Some key vector procedures:
 - `(vector val1 ... valn)`: Create a vector
 - `(make-vector length val)`: Make a vector of specified length, with duplicates of *val* as the contents.
 - `(vector-ref vector position)`: Extract a value from a vector.
 - `(vector-set! vector position newvalue)`: Change an element of a vector.
 - `(vector-length vector)`

Using Vectors for Palettes

- Why?
- Lets you more easily change the palette (using `vector-set!`).
- Much faster in rebuilding an image from indices
- Basic operations:
 - Create empty palette
 - Add a color to the palette
 - Find closest color in the palette
 - Given an index, get the corresponding color
- ...
- How?
 - Create fixed-size vector
 - Fill with special value (`color-transparent` in the original reading; updated to be `null`).

Lab

- Do the lab.

Copyright © 2007-8 Janet Davis, Matthew Kluber, and Samuel A. Rebelsky. (Selected materials copyright by John David Stone and Henry Walker and used by permission.) This material is based upon work partially supported by the National Science Foundation under Grant No. CCLI-0633090. Any opinions,

findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. This work is licensed under a Creative Commons Attribution-NonCommercial 2.5 License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/2.5/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.