ALTA: A BRDF analysis library

Laurent Belcour[†1,2]   Pascal Barla[2 ‡]   Romain Pacanowski[§3,2]

[1]Université de Montréal   [2]Inria Bordeaux Sud-Ouest   [3]LP2N-CNRS

Abstract
In this document, we introduce ALTA, a cross platform generic open-source library for Bidirectional Reflectance Distribution Function (BRDF) analysis. Among others, ALTA permits to estimate BRDF models parameters from measured data, to perform statistical analysis and also to export BRDF data models in a wide variety of formats.

1. Motivation

The Bidirectional Reflectance Distribution Function (BRDF) [Nic65] is a common way to represent material appearance in Computer Graphics (CG). Two reasons for their widespread adoption are the availability of measurements of BRDF data and efficient analytical models. However, BRDFs are four dimensional functionals that makes data analysis, storage and model fitting a complex and demanding task.

Providing a good representation for BRDFs and accurate measurements are challenges of modern CG. On the one side, mathematical models have to be representative of captured data while being efficient in terms of computation and sampling. On the other side, measurements of real objects need to be reproducible and must ensure that data values are reliable.

Recently, there have been a lot of development in the area of BRDF modeling [PCS*12, BSS*12, LKYU12, Bur12]. Unfortunately, these models have not been compared with each another. The only source of comparisons we have of BRDF models is through the near decade-old work of Ngan et al. [NDM05], which is now fairly outdated. Furthermore, the authors of the newly introduced techniques rarely compare evaluation and sampling performance.

Although some techniques exist for BRDF modeling [CPJ06, Bur], BRDF analysis and comparisons are performed using tools that are not shared within the research community or discontinued [HPBP12]. We introduce ALTA, a cross platform generic open-source BRDF analysis library

---

† laurent.belcour@umontreal.ca
‡ pascal.barla@inria.fr
§ romain.pacanowski@institutoptique.fr

under Mozilla Public License v2. The goal of this library is to provide a mean to compare and experiment with BRDF representations.The ALTA library is currently in an alpha stage but code source can be found at the following URL:

http://alta.gforge.inria.fr.

2. Overview of the Library

ALTA organizes BRDF analysis and modeling into atomic *programs* (coordinate transforms, sub-space integration, parameters estimation, *etc*) that are combined with *plugins*. Programs work on three types of plugins: *BRDFs*, *data*, and *fitters* (see Figure 1). Plugins can input or output files. BRDFs, data and fitters can be provided by users through dynamic libraries. This makes ALTA easily extensible. The complete list of ALTA current programs and features is detailled in Table 2. Programs are executed using command line instructions and can be chained to generate *scripts*. We provide a higher level interface to build scripts using XML files.

The list of currently supported programs are:

- **data conversions** (`brdf2data` and `data2data`) convert a BRDF or data file into another data format and output it. This can be useful to change parametrization or to densify data using data interpolation plugins.
- **function fitting** (`data2brdf`) computes the parameters of a BRDF model that matches the data given as input. A file containing the parameters is outputted.
- **function format conversion** (`brdf2brdf`) converts a BRDF model from a parameter file into another format such as GLSL shaders, matlab code or C/C++ code.
- **data or function moment computation** (`brdf2moments` and `data2moments`) output statistical information of a given BRDF or data file.
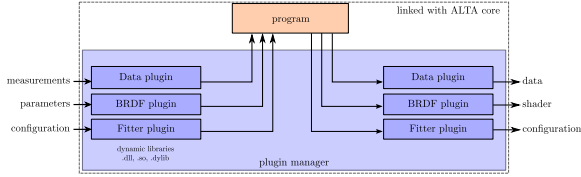
By chaining programs, we can perform complex scripts

Figure 1: *ALTA work-flow is decomposed using three elements: plugins, programs, and files. A plugin provides either a BRDF model, a data importer/exporter, or a fitting procedure. Plugins can be used as either input or output of programs. A program connects different plugins types to perform a specific action (fitting a BRDF model onto a data file, converting a data file into another format, etc). Files are used as inputs and outputs of the work-flow. The file format depends on the plugin used (binary file, C++ file, GLSL shader file, ...).*

on BRDF data and models. We provide below the example of the fitting `gold-metallic-paint` sample from the MERL database [MPBM03] with a BRDF model composed of a Lambertian diffuse component and a Beckmann specular lobe using the non-linear CERES solver from Google [AMO]. The fitting is then exported to a GLSL shader compatible with BRDF Explorer [Bur]:

```
$ ./data2brdf -input gold-met.dat -
output gold.brdf -func [libnonlin-
ear_function_diffuse.so, libnonlin-
ear_function_beckmann.so] -fitter libnon-
linear_fitter_ceres.so -max [0.8, 0.8, 0.01]
$ ./brdf2brdf -input gold.brdf -
output gold.glsl -export shader
```

Here we remove part of the data using the `-max [0.8, 0.8, 0.01]` argument. It permits to ensure fitting stability [NDM05]. This example works on a GNU/Linux operating system (OS) but other OS configurations (Microsoft's Windows and Apple's OSX) are supported.

Since writing command lines might be tedious and is platform dependent, we provide a high level mechanisms using XML-based scripts to overcome those issues. The example of fitting the gold metallic paint can be translated to a XML file and is shown in Listing 1.

```xml
<?xml version="1.0"?>
<alta>
  <action name="data2brdf">
    <!-- Input and output arguments of the action -->
    <input  name="gold-metallic.dat" />
    <output name="gold-metallic.brdf" />
    <parameter name="max" value="[0.8, 0.8, 0.01]" />

    <!-- Define the function to use -->
    <function name="nonlinear_function_diffuse" />
    <function name="nonlinear_function_beckmann" />

    <!-- Define the ftting procedure to use -->
    <plugin type="fitter" name="nonlinear_fitter_ceres" />
  </action>
```

```xml
  <action name="brdf2brdf">
    <!-- Input and output arguments of the action -->
    <input  name="gold-metallic.brdf" />
    <output name="gold-metallic-export.brdf" />

    <!-- Parameter -->
    <parameter name="export" value="explorer" />
  </action>
</alta>
```

Listing 1: *Example of a high-level XML script fitting the* `gold-metallic-paint` *from the MERL database [MPBM03] to a Beckmann + diffuse BRDF. the resulting BRDF file is then converted to a compatible BRDF Explorer [Bur] shader.*

We plan to extend it to high level, OS independent, scripting tools using the possibility of SWIG to provide Python, Octave, R, *etc* interfaces to ALTA. This will allow users to have a more direct access to the possibilities of ALTA, while permitting to share scripts with the rest of the community.

3. Use Cases: Fitting Algorithms

We have described the generic behavior of ALTA. In this section, we detail two specific and already present features of the library: multiple non-linear fitting algorithms and implementations, and rational BRDF models and fitters.

3.1. Non-linear fitting

Non-linear optimization methods can be used to find parameters of a BRDF model to approximate data. These techniques solve the following problem in a general way:

$$\text{argmin}_{\vec{p}} ||\rho_{\vec{p}}(\omega_i, \omega_o) - \text{data}(\omega_i, \omega_o)||,$$

where $\vec{p}$ is the parameter vector of the BRDF model ρ and $||\cdot||$ is a user chosen norm function. No assumption is made on the relationship between the BRDF model and its parameters. The norm is usually tied to a specific type of algorithm. For example, methods using gradient descents preferably use L^2 norms since the gradient of the error with respect to the parameter vector is easily computed.

Thanks to our plugin structure, we can compare the performance and quality of several fitting algorithms. By default, ALTA comes with four non-linear solvers: Google CERES solver, Eigen Levenberg-Marquardt solver, MIT NlOpt solver and CoinOr solver. By providing multiple fitting plugins, we let users choose of the norm to optimize and the best associated method for it. We show the results of one of ALTA non-linear fitting plugin in Figure 2.

3.2. Rational BRDF Approximation

We implemented Rational BRDF [PCS*12] as well as several ways to perform approximation of BRDF data using rational polynomials. We provide: a constrained least-square

(a) MERL   (b) Beckmann lobe fit
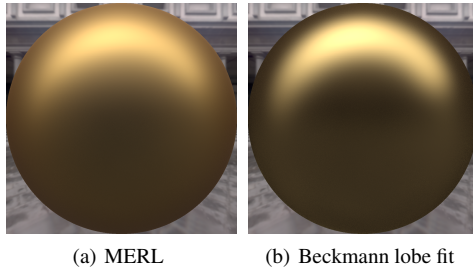
Figure 2: *We show the result of Listing 1 displayed using BRDF Explorer [Bur] on the left. The output of the script can be reused to add other components to the BRDF such as a Lafortune lobe to further reduce the error.*

(a) Acquired data   (b) Rational interpolation

Figure 3: *We compare the result of R-BRDF approximation with 62 samples (b) of the MERL* `gold-metallic-paint` *sample (a), converted to a 2D parametrization* (θ_h, θ_d), *where* $\phi_d = 0$. *To produce this image, the result of the fitting procedure was used as input to* `brdf2data` *program and the* `data-slice` *plugin was used for the output.*

approximation of data using Eigen, rational interpolation of vertical segments using quadratic programming (using either CGAL, Matlab or QuadProg++ library), and rational optimization of the L^∞ norm using the DCA algorithm. We detail the specificity of our rational interpolation of vertical segments.

Contrary to non-linear solvers, R-BRDF fitting does not optimize for the L^2 norm but rather interpolate intervals (vertical segments) around the measurement data values $p(u_i) \in [\underline{\rho}_i .. \bar{\rho}_i]$. This translates into a constraint matrix $\mathcal{A}$ and the optimization is done on the resulting coefficient vector's norm:

$$\min ||x||^2 \text{ such that } \mathcal{A}x \geq \delta ||x||,$$

where x is the concatenated vectors of coefficients of the rational function $p(u) = \frac{\sum_i x_i p_i(u)}{\sum_j x_j q_j(u)}$. We found that in practice, the use of the condition number δ of the constraints matrix $\mathcal{A}$ over-constrains the solution. Although it permits to avoid zero solutions, it severely impacts performance. Since the SVD of $\mathcal{A}$ needs to be calculated, its cost is non-negligible, we often use a constant value to avoid zero solution:
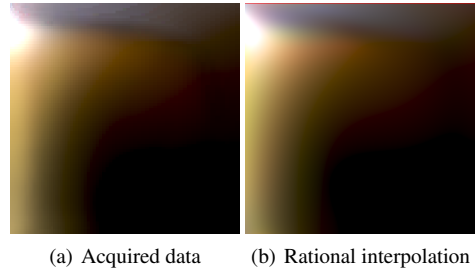
$$\mathcal{A}x \geq c ||x||,$$

where c can be a user defined parameter (1 by default).

We applied R-BRDF approximation to the `gold-metallic-paint` from the MERL database projected to a 2D parametrization (θ_h, θ_d), where $\phi_d = 0$ (Figure 3). Exporting the data to an image format is done within ALTA, using the *brdf2data* program.

4. Features List

In this section, we quickly describe other functionalities of the ALTA library (cf. Table 2 for the complete list of features.)

4.1. Data Functionalities

ALTA handles commonly used material data formats such as MERL binary format, and ASTM text format. We provide our own internal format which is text-based and compatible with tools such as GnuPlot or Matlab. We plan to incorporate a binary file format to handle large amounts of data.

In table 1. we list all currently available parametrizations that may be used conversion programs in conversion programs. Note that conversion from any parametrization to another is possible. Adding additional parametrizations using the plugin mechanism is currently not supported but planned for future releases.

Parametrization	Dimensions
Cartesian	6
Half-angle [Rus98]	4
Projected half-angles [LKYU12]	4
Stereographic projection	4
Barycentric parametrizations [SAS05]	2

Table 1: *List of available parametrizations in ALTA core with the associated dimensionality of the parametrization.*

It is possible to perform data interpolation and reconstruction using data plugins and the *data2data* program. We currently provide two interpolation plugins, one using inverse distance weighting for a k-nearest neighbor search and the other making use of Matlab linear, quadratic and cubic interpolation packages.

4.2. BRDF Models

We handle both parametric and non-parametric BRDF models through a simple interface. BRDF fitting can be exported to various formats such as GLSL shaders (compatible with BRDF Explorer), C++ code, or Matlab code. We currently

handle the simple cosine model [Bli77], the extended cosine model [LFTG97], Gaussian models [BS87, War92], the ABC model [LKYU12], as well as different forms of shadowing [Sch94, Smi67], Fresnel [Sch94] terms and Rational BRDFs [PCS*12]. Of course our set of models is currently very small compared to the variety of models used in computer graphics but our plugin system permits to easily to extend this list and plans are made to provide the most complete set of BRDF models for comparison.

4.3. Statistical Information

Data analysis can be beneficial prior to any operation (conversion, fitting, etc). We thus include programs that compute raw moments from either a BRDF model (`brdf2moments`) or BRDF data (`data2moments`). Moments are statistical measurements that characterize the shape of a distribution, and are given by:

$$\kappa_k = \sum_{i \le k} a_i \mu_i, \text{ where } \mu_k = \int_D x^k f(x)\mathrm{d}x. \quad (1)$$

We also provide for the computation of cumulants, which are obtained as combinations of raw moments [MK09]. They are often most interesting as they describe centered and scaled statistics of a distribution. Cumulants at order 2 correspond to (co-)variance and measure the spread of the distribution. Cumulants at order 3 are related to (co-)skewness that informs on the (a)symmetry of the distribution. Cumulants at order 3 are related to (co-)kurtosis that characterizes the 'peakedness' of the distribution.

They may be used in a variety of ways. For instance, close-to-zero cumulants are informative of (anti-)symmetries or potentially separable distributions.

5. Mid-and long-term goals

Our goal is to obtain a complete overview of BRDF models accuracy to represent measurement databases available in the public domain. Although we already support most of the formats used for BRDF data and a fair lot of fitters, it requires providing a plugin for any of the BRDF model from the literature. We would like to go further and create a comparison website where researchers could upload BRDF models, fitting plugins, or BRDF data and automatically see the fitting comparisons on an automatically generated web page.

Another goal is to extend the library interface to include sampling methods for BRDFs. This will permit to perform in-depth comparisons of BRDF models performance and sampling efficiency.

We would like to extend ALTA modularity and for instance enable users to specify a transformation on BRDF values during the fitting procedure. As shown by [LKYU12], BRDF value transformations allow to improve on fitting performance.

We would also like the research and industrial community on materials to take part into the development of ALTA. This is why we distribute it with an open-source license compatible with commercial usage. Users can write scripts and share them. They can also write and share plugins for new BRDF models, data formats or fitting procedures. Furthermore, extending ALTA set of programs is possible since the core of our library is Open Source and documented. We plan to further improve ALTA capabilities to support high dimensional BRDF data and models such as polarized BRDF, spatially-varying BRDF and even BSSRDF.

References

[AMO] AGARWAL S., MIERLE K., OTHERS: CERES solver. 2

[Bli77] BLINN J. F.: Models of light reflection for computer synthesized pictures. *ACM Computer Graphics (Proceedings of SIGGRAPH 1977) 11*, 2 (Aug. 1977), 192–198. 4, 5

[BS87] BECKMANN P., SPIZZICHINO A.: *Scattering of Electromagnetic Waves from Rough Surfaces.* Artech House Publishers, 1987. 4, 5

[BSS*12] BAGHER M., SOLER C., SUBR K., BELCOUR L., HOLZSCHUCH N.: Interactive rendering of acquired materials on dynamic geometry using bandwidth prediction. In *Proceedings of ACM Symposium on Interactive 3D Graphics and Games - I3D 2012* (Mar. 2012), ACM, pp. 127–134. 1

[Bur] BURLEY B.: BRDF Explorer. 1, 2, 3

[Bur12] BURLEY B.: *Physically-Based Shading at Disney.* Tech. rep., Walt Disney Animation Studios, 2012. 1

[CPJ06] COLBERT M., PATTANAIK S., JAROSLAV K.: BRDF-Shop: An Artistic Tool for Creating Physically Correct BRDFs. *IEEE Computer Graphics and Applications 26*, 1 (2006), 30—-36. 1

[HPBP12] HERRANZ A. F., PATTANAIK S., BOSCH C., PUEYO X.: BRDF Lab, 2012. 1

[LFTG97] LAFORTUNE E., FOO S.-C., TORRANCE K., GREENBERG D.: Non-linear approximation of reflectance functions. In *SIGGRAPH 1997* (Aug. 1997), ACM Press, pp. 117–126. 4, 5

[LKYU12] LÖW J., KRONANDER J., YNNERMAN A., UNGER J.: BRDF models for accurate and efficient rendering of glossy surfaces. *ACM Transactions on Graphics 31*, 1 (Jan. 2012), 1–14. 1, 3, 4, 5

[MK09] MCCULLAGH P., KOLASSA J.: Cumulants. *Scholarpedia 4*, 3 (2009), 4699. revision #91184. 4

[MPBM03] MATUSIK W., PFISTER H., BRAND M., MCMILLAN L.: A data-driven reflectance model. *ACM Transactions on Graphics 22*, 3 (2003), 759–769. 2

[NDM05] NGAN A., DURAND F., MATUSIK W.: Experimental Analysis of BRDF Models. In *Eurographics Workshop on Rendering* (2005). 1, 2

[Nic65] NICODEMUS F. E.: Directional Reflectance and Emissivity of an Opaque Surface. *Applied Optics 4*, 7 (July 1965), 767–773. 1

[PCS*12] PACANOWSKI R., CELIS O. S., SCHLICK C., GRANIER X., POULIN P., CUYT A.: Rational BRDF. *IEEE transactions on visualization and computer graphics 18*, 11 (Feb. 2012), 1824–1835. 1, 2, 4

[Rus98] RUSINKIEWICZ S.: A New Change of Variables for Efficient BRDF Representation. In *Rendering Techniques (Proceedings of the Eurographics Symposium on Rendering 1998)*, Eurographics Book Series. Springer-Verlag, 1998. 3

[SAS05] STARK M. M., ARVO J., SMITS B.: Barycentric parameterizations for isotropic BRDFs. *IEEE transactions on visualization and computer graphics 11*, 2 (Jan. 2005), 126–38. 3

[Sch94] SCHLICK C.: An Inexpensive BRDF Model for Physically-based Rendering. *Computer Graphics Forum 13*, 3 (Aug. 1994), 233–246. 4, 5

[Smi67] SMITH B.: Geometrical Shadowing of a Random Rough Surface. *IEEE Transactions on Antennas and Propagation 15*, 5 (1967), 668–671. 4, 5

[War92] WARD G. J.: Measuring and modeling anisotropic reflection. *ACM Computer Graphics (Proceedings of ACM SIGGRAPH 1992) 26*, 2 (1992), 265–272. 4, 5

Appendix A: Table of ALTA's features

programs	plugins		
	datas	**BRDFs**	**fitters**
• `data2data` converts a data file into another format useful to change parametrization. • `data2brdf` fits a BRDF model to match data file, save to a BRDF file. • `brdf2data` converts a BRDF file to a data file. • `brdf2brdf` exports a BRDF file to another format (C++, Matlab, shader). • `data2moments` estimates the moments of a data file. • `brdf2moments` estimates the moments of a BRDF file.	• `data_alta`: our own ASCII file format, compatible with gnuplot and Matlab. • `data_merl`: MERL binary file format • `data_slice`: (θ_h, θ_d) BRDF slices stored in EXR image format. • `data_astm` *American Society for Testing and Materials* Standard E1392-96 ASCII file format. • `data_interpolant_matlab` provides continuous data from a sparse set using Matlab interpolation packages. • `data_interpolant_flann` provides continuous data from a sparse set using k-nearest neighbors interpolation.	We prodive two rational BRDF basis to work with the rational fitters: • A rational BRDF plugin using a `legendre` basis, • ang a rational BRDF plugin using a `chebychev` basis. We prodive BRDF to work with the nonlinear fitters: • `beckmann`: microfacet based Beckamnn lobe [BS87]. • `abc`: ABC model [LKYU12]. • `blinn`: power of cosine lobe [Bli77]. • `lafortune`: extended cosine lobe [LFTG97]. • `ward`: anisotropic Gaussian BRDF model [War92]. • `schlick`: rational approximation of the Fresnel term [Sch94]. • `smith`: shadowing from a Gaussian microfacet surface [Smi67].	• `rational_quadprog`: rational BRDF interpolation of vertical segments using QuadProg++. • `rational_matlab`: rational BRDF interpolation of vertical segments using Matlab. • `rational_cgal`: rational BRDF interpolation of vertical segments using CGAL. • `rational_dca`: L^∞ optimization of a rational BRDF approximation. • `rational_parallel`: parallel rational BRDF interpolation of vertical segments using QuadProg++ and OpenMP. • `rational_leastsquare`: rational BRDF approximation using a constrained LS. • `nonlinear_ceres`: nonlinear BRDF approximation using CERES solver. • `nonlinear_eigen`: nonlinear BRDF approximation using EIGEN solver.. • `nonlinear_ipopt`: nonlinear BRDF approximation using IpOpt solver. • `nonlinear_nlopt`: nonlinear BRDF approximation using NlOpt solver.

Table 2: *The complete list of features present in the ALTA library.*